**2023 12 22**

***// std::chrono:duration***

**using** **namespace** std**;**

**using** **namespace** std**::**chrono**;**

***// count()***

**int** main**()**

**{**

milliseconds ms1 **{** 321313 **};**

**auto** val **=** ms1**.**count**();** ***// 321313***

**}**

***// duration fonksiyonlar***

**int** main**()**

**{**

***// nanoseconds***

**auto** dr **=** milliseconds**{**456**}** **+** nanoseconds**{**6'423'123**}** **+** seconds **{** 2 **};**

cout **<<** dr**.**count**();**

milliseconds ms **{**2131**};**

ms **=** 45**;** ***// örtülü dönüşüm syntax hatası***

**int** ival **=** ms**;** ***// syntax hatası***

**}**

**int** main**()**

**{**

milliseconds ms **{**3123**};**

microseconds us **{**4'434'543**};**

us **=** ms**;** ***// geçerli***

ms **=** us**;** ***// geçersiz (daha ince türden daha kaba türe dönüşüm syntax hatası olr)***

**}**

**using** halfsec **=** duration**<double,** ratio**<**1**,** 2**>>;**

**int** main**()**

**{**

halfsec hs **=** ms**;** ***// geçerli***

**}**

**int** main**()**

**{**

cout **<<** milliseconds**{** 345 **}** **<<** "\n"**;** ***// 345ms --C++20 ile geldi***

**}**

**UDL (user-defined literals)**

***// operator""ms(823) --823ms operator overloading***

**int** main**()**

**{**

**auto** dur **=** 345ms**;**

**constexpr** **auto** dur **=** 345ms**;**

**constexpr** **auto** dur1 **=** **operator**""ms**(**345**);**

**auto** val **=** 567ms **+** 3457us **+** 65423ns**;**

**}**

**int** main**()**

**{**

"mustafa"s ***// string sınıfından***

**operator**""s**(**"alican"**,** 6**);** ***// "alican"s***

**}**

***operator""\_x(long long double);***

***operator""\_y(unsigned long long);***

***operator""\_z(char);***

***operator""\_k(const char\*, size\_t);***

***cooked 569s 569'u doğrudan operator fonksiyonuna gönderiyorum***

***uncooked 21312x bir yazı olarak gönderiyorsak null karakter görene kadar okuyosak***

***// UDL oluşturma (namespace almak daha mantıklı)***

**constexpr** **double** **operator**""\_m**(long** **double** val**)**

**{**

**return** **static\_cast<double>(**val**);**

**}**

**constexpr** **double** **operator**""\_cm**(long** **double** val**)**

**{**

**return** **static\_cast<double>(**val **/** 100**);**

**}**

**constexpr** **double** **operator**""\_mm**(long** **double** val**)**

**{**

**return** **static\_cast<double>(**val **/** 1000**);**

**}**

**constexpr** **double** **operator**""\_km**(long** **double** val**)**

**{**

**return** **static\_cast<double>(**val **\*** 1000**);**

**}**

**int** main**()**

**{**

**constexpr** **auto** x **=** 4.5\_m ***// double***

**constexpr** **double** y **=** **operator**""\_m**(**4.5657**);**

**constexpr** **auto** distance **=** 3.5\_m **+** 876.35\_cm **+** 1234.12\_mm **+** 0.0812\_km**;**

cout **<<** distance **<<** " metre\n"**;**

**}**

**constexpr** std**::size\_t** **operator**""\_KB**(unsigned** **long** **long** size**)**

**{**

**return** **static\_cast<**std**::size\_t>(**size **\*** 1'024**);**

**}**

**constexpr** std**::size\_t** **operator**""\_MB**(unsigned** **long** **long** size**)**

**{**

**return** **static\_cast<**std**::size\_t>(**size **\*** 1'024 **\*** 1'024**);**

**}**

**constexpr** std**::size\_t** **operator**""\_GB**(unsigned** **long** **long** size**)**

**{**

**return** **static\_cast<**std**::size\_t>(**size **\*** 1'024 **\*** 1'024 **\*** 1'024**);**

**}**

**int** main**()**

**{**

std**::**array**<char,** 12\_KB**>** buf1**{};**

std**::**array**<char,** 1\_MB**>** buf2**{};**

**}**

**constexpr** **int** **operator**""\_i**(char** c**)**

**{**

**return** c**;**

**}**

**int** main**()**

**{**

**using** **namespace** std**;**

cout **<<** 'A' **<<** "\n"**;** ***// A***

cout **<<** 'A'\_i **<<** "\n"**;** ***// 65***

**}**

**unsigned** **constexpr** **int** **operator**""\_b**(const** **char\*** p**)**

**{**

**using** result**{};**

**while** **(\***p**)**

**{**

**char** digit **=** **\***p**;**

**if** **(**digit **!=** '1' **&&** digit **!=** '0'**)**

**{**

**throw** std**::**runtime\_error**{**"invalid ch : "s **+** digit**}**İ

**}**

result **=** result **\*** 2 **(**digit **-** '0'**);**

**++**p**;**

**}**

**return** result**;**

**}**

**int** main**()**

**{**

**using** **namespace** std**;**

**auto** val **=** 1010101\_b**;**

cout **<<** val **<<** "\n"**;**

**}**

**class** Kilogram

**{**

**public:**

**class** prevent\_usage**{};**

Kilogram**(**prevent\_usage**,** **double** val**)** **:** mweigth**{** val **}** **{}**

**friend** **constexpr** Kilogram **operator+(const** Kilogram**&** x**,** **const** Kilogram**&** y**)**

**{**

**return** Kilogram**{**Kilogram**::**prevent\_usage**{},** x**.**mweigth **+** y**.**mweigth**};**

**}**

**private:**

**double** mweigth**;**

**};**

**constexpr** Kilogram **operator**""\_kg**(long** **double** x**)**

**{**

**return** Kilogram**{** Kilogram**::**prevent\_usage**{},** **static\_cast<double>(**x**)};**

**}**

**constexpr** Kilogram **operator**""\_gr**(long** **double** x**)**

**{**

**return** Kilogram**{** Kilogram**::**prevent\_usage**{},** **static\_cast<double>(**x **/** 1000**)};**

**}**

**constexpr** Kilogram **operator**""\_mg**(long** **double** x**)**

**{**

**return** Kilogram**{** Kilogram**::**prevent\_usage**{},** **static\_cast<double>(**x **/** 1000 **/** 1000**)};**

**}**

**int** main**()**

**{**

**constexpr** Kilogram kg1 **=** 5.6\_kg**;**

**constexpr** **auto** weigth **=** 4.5\_kg **+** 1234.87\_gr **+** 345'434.55\_mg**;**

**}**

***// std::chrono (duration\_cast)***

**int** main**()**

**{**

**using** **namespace** std**::**chrono**;**

**auto** ms **==** 75624ms**;**

seconds s **=** ms**;** ***// sytantx hataso***

seconds s**(**ms**.**count**()** **/** 1000**);** ***// geçerli***

duration\_cast**<**seconds**>(**ms**);** ***// veri kaybı olur***

cout **<<** sec**.**count**()** **<<** "\n"**;** ***// 75***

**}**

**int** main**()**

**{**

**long** **long** val**;**

std**::**cout **<<** "milisaniye olarak degeri girin : "**;**

cin **>>** val**;**

milliseconds ms**{**val**};**

hours hrs **=** duration\_cast**<**hours**>(**ms**);**

minutes mins **=** duration\_cast**<**minutes**>(**ms **%** 1h**);**

seconds sec **=** duration\_cast**<**seconds**>(**ms **%** 1min**);**

**if** **(**hrs**.**count**())**

cout **<<** hrs**.**count**()** **<<** " saat"**;**

**if** **(**mins**.**count**())**

cout **<<** mins**.**count**()** **<<** " dakika"**;**

**if** **(**sec**.**count**())**

cout **<<** sec**.**count**()** **<<** "mili saniye"\n";

**}**

***// yuvarlama işlemleri***

**int** main**()**

**{**

**auto** ms **=** 923'879ms**;**

cout **<<** duration\_cast**<**seconds**>(**ms**)** **<<** "\n"**;** ***// 923***

cout **<<** floor**<**seconds**>(**ms**)** **<<** "\n"**;** ***// 923***

cout **<<** ceil**<**seconds**>(**ms**)** **<<** "\n"**;** ***// 924***

cout **<<** round**<**seconds**>(**ms**)** **<<** "\n"**;** ***// 924***

**}**

**template** **<typename** Rep**,** **typename** Period**>**

std**::**ostream**&** **operator<<(**std**::**ostream**&** os**,** **const** std**::**chrono**::**duration**<**Rep**,** Period**>&** dur**)**

**{**

**return** os **<<** dur**.**count**()** **<<** " \* " **<<** Period**::**num **<<** " / " **<<** Period**::**den **<<**

**}**

**using** tensec **=** std**::**chrono**::**duration**<int,** std**::**ratio**<**10**>>**

**int** main**()**

**{**

**using** **namespace** std**;**

**using** **namespace** std**::**chrono**;**

cout **<<** 3456ms**;** ***// 3456 \* ( 1 / 1000)***

cout **<<** tensec **{**6512**}** **<<** "\n"**;**

**}**

***// system\_clock***

**int** main**()**

**{**

**using** **namespace** std**;**

**using** **namespace** std**::**chrono**;**

system\_clock**::**time\_point ***// time\_point<system\_clock, system\_clock::duration>***

**auto** tp1 **=** system\_clock**::**now**();**

**auto** tp2 **=** system\_clock**::**now**();**

**auto** diff **=** tp2 **-** tp1**;**

**}**

***// süre ölçümü***

std**::**vector**<size\_t>** get\_sorted\_vector**(**std**::size\_t** n**)**

**{**

std**::**vector**<size\_t** **>** vec**(**n**);**

std**::**generate\_n**(**vec**.**begin**(),** n**,** rand**);**

std**::**sort**(**vec**.**begin**(),** vec**.**end**());**

**return** vec**;**

**}**

**int** main**()**

**{**

**using** **namespace** std**;**

**using** **namespace** std**::**chrono**;**

**auto** tp\_start **=** steady\_clock**::**now**();**

**auto** vec **=** get\_sorted\_vector**(**450'000**);**

**auto** tp\_end **=** stready\_clock**::**now**();**

cout **<<** duration\_cast**<**milliseconds**>(**tp\_end **-** tp\_start**).**count**()** **<<** " milisaniye\n"**;**

cout **<<** duration\_cast**<double>(**tp\_end **-** tp\_start**).**count**()** **<<** " saniye\n"**;**

cout **<<** duration\_cast**<double,** micro**>(**tp\_end **-** tp\_start**).**count**()** **<<** " micro saniye\n"**;**

cout **<<** "vec.size()" **<<** vec**.**size**()** **<<** "\n"**;**

**}**

**int** main**()**

**{**

**using** **namespace** std**;**

**using** **namespace** std**::**chrono**;**

**auto** tp **=** system\_clock**::**now**();**

time t ct\_time **==** system\_clock**::**to\_time\_t**(**tp**);**

cout **<<** ctime**(&**c\_time**)** **<<** "\n"**;**

**}**