**2023 12 29**

**std::any**

***// std::any***

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

**{**

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

any a**(**45**);**

cout **<<** a**.**type**().**name**()** **<<** '\n'**;**

**if** **(**a**.**type**()** **==** **typeid(int))**

**{**

**auto** ai **=** any\_cast**<int>(**a**);** ***// bad\_any\_cast throw eder eğer int değilse***

**}**

**int** **&**r **=** any\_cast**<int&>(**a**);**

**++**r**;**

cout **<<** any\_cast**<int>(**a**);** ***// 46***

**}**

***// reset()***

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

**{**

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

any a **=** "emrecan suster"**;**

cout **<<** **(**a**.**has\_value**()** **?** "dolu" **:** "bos"**)** **<<** "\n"**;** ***// dolu***

a**.**reset**();**

cout **<<** **(**a**.**has\_value**()** **?** "dolu" **:** "bos"**)** **<<** "\n"**;** ***// bos***

**}**

**void** **\*operator** **new(**std**::size\_t** n**)**

**{**

std**::**cout **<<** "operator new called... n = " **<<** n **<<** "\n"**;**

**return** std**::**malloc**(**n**);**

**}**

**void** **operator** **delete(void\*** vp**)**

**{**

**if** **(**vp**)**

std**::**free**(**vp**);**

**}**

**struct** Nec **{**

**unsigned** **char** buffer**[**1024**]{};**

**};**

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

**{**

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

***// size derleyiciden derleyiciye değişir***

std**::**cout **<<** "sizeof(any) = " **<<** **sizeof(**any**)** **<<** "\n"**;** ***// 40***

std**::**cout **<<** "sizeof(type\_info) = " **<<** **sizeof(**type\_info**)** **<<** "\n"**;** ***// 12***

any ax **=** 12**;**

ax **=** 4.5**;**

ax **=** bitset**<**128**>{};**

ax **=** Nec**{};** ***// operator new çağrılır***

**}**

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

**{**

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

**int** arr**[**100**]{};**

any a **=** arr**;**

std**::**cout **<<** "sizeof(arr) = " **<<** **sizeof(**arr**)** **<<** "\n"**;** ***// 400***

**if** **(** a**.**type**()** **==** **typeid(int\*))**

std**::**cout **<<** "pointer tutuyor\n"**;** ***// pointer tutuyor çıkar***

**else** **if(**a**.**type**()** **==** **typeid(int[**100**]))**

std**::**cout **<<** "dizi tutuyor\n"**;**

**}**

***// emplace ve make\_any***

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

**{**

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

any a**;**

a **=** 4567**;**

a**.**emplace**<**Date**>(**3**,** 5**,** 1987**);**

make\_any**<**Date**>(**2**,** 5**,** 1987**);**

**}**

***// Kullanım Senaryosu***

**using** tvp **=** std**::**pair**<**std**::**string**,** std**::**any**>**

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

**{**

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

vector**<**tvp**>** vec**;**

vec**.**emplace\_back**(**"name"**,** "tamer dundar"s**);**

vec**.**emplace\_back**(**"birth\_year"**,** 1994**);**

vec**.**emplace\_back**(**"month"**,** 11**);**

vec**.**emplace\_back**(**"price"**,** 59.97**);**

vec**.**emplace\_back**(**"town"**,** "mugla"s**);**

vec**.**emplace\_back**(**"gender"**,** "male"s**);**

**for** **(const** **auto&[**property**,** value**]** **:** vec**)**

**{**

**if** **(**value**.**type**()** **==** **typeid(int))**

**{**

cout **<<** property **<<** " " **<<** any\_cast**<int>(**value**)** **<<** "\n"**;**

**}**

**else** **if** **(**value**.**type**()** **==** **typeid(**string**))**

**{**

cout **<<** property **<<** " " **<<** any\_cast**<**string**>(**value**)** **<<** "\n"**;**

**}**

**else** **if** **(**value**.**type**()** **==** **typeid(double))**

**{**

cout **<<** property **<<** " " **<<** any\_cast**<double>(**value**)** **<<** "\n"**;**

**}**

**}**

**}**

***// in\_place\_type***

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

**{**

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

any ax**{**in\_place\_type**<**Date**>,** 1**,** 1**,** 2024**};**

**}**

**Std::random**

***// std::random***

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

**{**

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

default\_random\_engine ***// std::mt19937***

cout **<<** **(typeid(**mt19937**)** **==** **typeid(**default\_random\_engine**))** **<<** "\n"**;**

mt19937\_64 ***// 64 bitlik üretir***

cout **<<** mt19937**::**default\_seed **<<** "\n"**;**

**}**

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

**{**

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

mt19937 eng**{**123**};**

**for** **(int** i **=** 0**;** i **<** 10000 **-** 1**;** **++**i**)**

**(void)**eng**();** ***// bütün derleyicilerden aynı olmak zorunda***

cout **<<** "min = " **<<** mt19937**::**min**()** **<<** "\n"**;**

cout **<<** "max = " **<<** mt19937**::**max**()** **<<** "\n"**;**

**}**

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

**{**

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

***// kopyalama yapmak çok maliyetli***

std**::**cout **<<** "sizeof(mt19937) = " **<<** **sizeof(**mt19937**)** **<<** "\n"**;** ***// 5000***

**}**

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

**{**

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

mt19937 eng**();**

***//// eng böyle kullanırsak kopyalama olur***

generate\_n**(**vec**.**begin**(),** 10'000**,** eng**);**

***// böyle kullanmnımlaı***

generate\_n**(**vec**.**begin**(),** 10'000**,** ref**(**eng**));**

**}**

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

**{**

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

mt19937 e1**(**32**);**

mt19937 e2**(**32**);**

mt19937 e3**(**56**);**

cout **<<** **(**e1 **==** e2**)** **<<** "\n"**;** ***// true***

cout **<<** **(**e1 **==** e3**)** **<<** "\n"**;** ***// false***

***// rastgele sayı üretince state değişir.***

**auto** val **=** e1**();**

**++**val**;**

cout **<<** **(**e1 **==** e2**)** **<<** "\n"**;** ***// false***

val **=** e2**();**

cout **<<** **(**e1 **==** e2**)** **<<** "\n"**;** ***// true***

**}**

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

**{**

mt19937 eng**{**123**};**

**for** **(int** i **=** 0**;** i **<** 10**;** **++**i**)** **{**

**(void)**eng**();**

**}**

stringstream ss**;**

ss **<<** eng**;**

mt19937 eng2**;**

ss **>>** eng2**;** ***// state kopyalamış olduk***

**}**

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

**{**

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

***// 10 - 14 arasında değerler veririr uniform olarak***

uniform\_int\_distribution**<int>** dist**{** 10**,** 14 **};**

mt19937 eng**;**

**for** **(int** i **=** 0**;** i **<** 100**;** **++**i**){**

cout **<<** dist**(**eng**)** **<<** ' '**;**

**}**

**}**

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

**{**

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

map**<int,** **int>** cmap**;**

uniform\_int\_distribution dis**{**0 **,** 20**};**

**for** **(int** i **=** 0**;** i **<** 20'000'000**;** **++**i**)**

**{**

**++**cmap**[**dist**(**eng**)];** ***// rastgele sayılar birbirine yakın sayıda üretilir***

**}**

**}**

***// uniform\_real\_distribution***

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

**{**

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

mt19937 eng**;**

uniform\_real\_distribution dist**{** 5.6**,** 6.1**};**

**for** **(int** i **=** 0**;** i **<** 100**;** **++**i**)**

cout **<<** dist**(**eng**)** **<<** "\n"**;**

**}**

***// normal\_distribution***

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

**{**

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

mt19937 eng**;**

normal\_distribution**<double>** dist**(**50**,** 5.**);**

**}**