**C++ KURS İÇERİĞİ**

* **2023.07.24**
  1. **C in C++**
* **2023.07.26**
  1. **Const**
     + **top level const**
     + **low level const**
  2. **C in C++**
  3. **Initilization**
* **2023.07.28**
  1. **Reference Semantics**
     + **L value reference**
  2. **Value Category**
* **2023.07.31**
  1. **Reference Semantics**
     + **R value reference**
  2. **Type Deduction**
     + **Auto type deduction**
  3. **Reference Collapsing**
* **2023.08.02**
  1. **Reference Collapsing**
  2. **Universal Reference**
  3. **Type Deduction**
     + **Decltype specifier**
  4. **Unevaluated Context**
  5. **Function Default Arguman**
  6. **Constexpr**
* **2023.08.04**
  1. **Constexpr**
  2. **Constexpr functions**
  3. **ODR (One Definition Rule)**
  4. **Inline Expansion**
  5. **Enumaration Types**
* **2023.08.07**
  1. **Scoped Enum**
  2. **Type-Cast Operators**
     + **static\_cast**
     + **const \_cast**
     + **reinterpret\_cast**
     + **dynamic\_cast**
* **2023.08.09**
  1. **Function Overloading**
* **2023.08.11**
  1. **Classes**
* **2023.08.14**
  1. **Classes**
     + **This keyword**
     + **Mutable**
* **2023.08.16**
  1. **Classes**
     + **Constructor**
     + **Destructor**
     + **Special Member Functions**
     + **Global Class Objects**
     + **Constructor Initializer List**
* **2023.08.18**
  1. **Classes**
     + **Delete bildirimi**
     + **Class Special Functions**
     + **Aggregate Class**
     + **Copy Constructor**
* **2023.08.21**
  1. **Classes**
     + **Copy Constructor**
     + **Copy Assignment**
     + **Move Constructor**
* **2023.08.23**
  1. **Class special member functions**
  2. **temporary objects C++17**
  3. **explicit ctor**
  4. **conversion ctor**
  5. **moved-from state (taşınmış nesne durumu)**
* **2023.08.25**
  1. **copy elision**
  2. **C++17 (mandatory copy elision)**
  3. **Return Value Optimization RVO**
  4. **NRVO (Named Return Value Optimization)**
  5. **new ifadeleri**
  6. **delete ifadeleri**
  7. **operator new fonksiyonları**
  8. **operator delete fonksiyonları**
  9. **sınıfların static veri elemanları**
* **2023.08.28**
  1. **Sınıfın static data member**
     + **İncomplete type**
     + **Header-Only Library**
  2. **Sınıfı static member functions**
     + **Named Ctor**
     + **Singleton Pattern**
     + **Mayer’s Singleton**
* **2023.09.01**
  1. **Delegating Ctor**
  2. **Friend Bildirimleri**
  3. **Operator Overloading**
* **2023.09.04**
  1. **Global Operator Function vs Member Operator Function**
  2. **Operator Overloading’te Function Overloading**
  3. **Neden member operator function ve global operator function ayrı ayrı var?**
  4. **Const Correctness & Operator Functions**
  5. **Operator Overloading ile Value Category İlişkisi**
  6. **Arrow Operator Overloading Fonksiyon**
* **2023.09.06**
  1. **Operator overload ++ ve –**
  2. **Operator overload + ve –**
  3. **Operator []**
* **2023.09.08**
  1. **Arrow operator overloading**
  2. **Overloading function call operator ()**
  3. **Tür dönüşüm operatörleri**
  4. **Conversion Ctor vs Operator Conversion**
  5. **Operator bool**
* **2023.09.11**
  1. **extern "C" bildirimi**
  2. **Composition (has a relation-ship)**
  3. **Copy Ctor && Move ctor && Copy Assignment && Move Assignment**
  4. **Reference Qualifiers**
* **2023.09.13**
  1. **Namespace ( İsim alanları )**
  2. **Using declaration**
  3. **using namespace directive**
  4. **ADL (argument dependent lookup) argümana bağlı isim arama**
* **2023.09.15**
  1. **Operator Overloading Enum**
  2. **Nested classes**
  3. **Design Pattern (pimpl idiom)**
  4. **Inheritance ( Kalıtım)**
* **2023.09.18**
  1. **Inheritance ( Kalıtım)**
  2. **Kalıtım ve İsim Arama**
  3. **Kalıtımda Using Bildirimi**
  4. **Kalıtımda special fonksiyonların durumları**
* **2023.09.20**
  1. **dynamic binding or late binding**
  2. **Non virtual Interface (NVI)**
  3. **Override Keyword**
  4. **Virtual Functions -Multi Level Inheritance**
  5. **Default Arguman in Virtual Functions**
  6. **Overloading Functions – Inheritance**
  7. **Virtual Ctor C++ (Clone idiom)**
* **2023.09.22**
  1. **Virtual DTOR**
  2. **Virtual Function Table Pointer (VPTR)**
  3. **Inheritanced Ctor**
  4. **Covariance or Variant Return Type**
  5. **NVI (non-virtual interface)**
  6. **Final Contextual Keyword**
     + **Final Class**
     + **Final Override**
* **2023.09.25**
  1. **Private Inheritance**
  2. **Restricted Polymporhsim**
  3. **Protected Inheritance (multi-level inheritance daha sık kullanılır)**
  4. **Multiple Inheritance**
  5. **Diamond Formation**
  6. **RTTI ( Run Time Type Information)**
* **2023.09.27**
  1. **RTTI ( Run Time Type Information)**
     + **Dynamic Cast**
     + **Typeid**
       - **Typeinfo**
  2. **Template Method**
* **2023.09.29**
  1. **std::initializer\_list**
  2. **std::string**
  3. **std::string CTOR**
  4. **Small String Optimizasyon (SSO)**
* **2023.10.02**
  1. **std::string elemanlara erişme**
  2. **Range-Based For Loop**
  3. **std::string atama işlemleri işlemleri**
  4. **std::string elemanları silme işlemleri**
  5. **std::string yazı ekleme işlemleri**
  6. **std::string arama işlemleri**
  7. **std::string yazıyı değiştirme işlemleri**
  8. **std::string karşılaştırma işlemleri**
* **2023.10.11**
  1. **std::string\_view**
  2. **std::string tür dönüşümleri**
  3. **Excaption Handling**
* **2023.10.13**
  1. **Excaption Handling**
  2. **Stack Unwinding**
  3. **Excaption Rethrow**
  4. **Excaption Handling in CTOR**
  5. **Exception Gurantee(s)**
* **2023.10.16**
  1. **Noexcept**
     + **noexcept specifier**
     + **noexcept operator**
  2. **Generic Programlama (Templates)**
* **2023.10.18**
  1. **Template Argument Deduction**
  2. **Universal Reference (Forwarding Reference)**
  3. **Trailing Return Type**
  4. **Auto Return Type**
  5. **Overloading in Template Functions**
* **2023.10.20**
  1. **Template Specialization**
  2. **Template Fonksiyon tanımlama**
  3. **Non-Type Parameter Template**
  4. **Default Template Arguman**
  5. **Explicit Specialization**
* **2023.10.23**
  1. **Explicit Specialization Template**
  2. **Partial Specialization Template**
  3. **Alias Template**
  4. **Variadic Template**
  5. **Recursive Instantiation**
  6. **Fold Expressions ( katlama ifadeleri)**
* **2023.10.25**
  1. **Template Value Type**
  2. **Static if**
  3. **Standart Template Library**
     + **Iterators**
     + **Iterator Category**
* **2023.10.27**
  1. **Algorithms**
     + **Std::copy**
     + **Std::copy\_if**
     + **Std::count\_if**
     + **Std::find**
  2. **Iterators**
     + **Const iterator (const\_iterator)**
     + **Reverse\_iterator**
* **2023.10.30**
  1. **Reverse Iterator**
  2. **Std::back\_inserter**
  3. **Std::front\_inserter**
  4. **Std::advance**
  5. **Std::distance**
  6. **Std::next**
  7. **Std::prev**
  8. **Std::iter\_swap**
  9. **Algorithms**
     + **Std::find\_if**
     + **Std::transform**
* **2023.11.01**
  1. **Lambda expression**
     + **Mutable lambda expression**
     + **Capture all by reference**
     + **Positive lambda expression**
     + **Noexpect lambda expression**
     + **Constexpr lambda expression**
* **2023.11.03**
  1. **Algorithms**
     + **Std::for\_each**
     + **Std::any\_of**
     + **Std::all\_of**
     + **Std::none\_of**
     + **Std::replace**
     + **Std::reverse\_copy**
     + **Std::remove\_copy\_if**
     + **Std::replace\_copy**
     + **Std::replace\_copy\_if**
  2. **Container**
     + **Std::vector**
* **2023.11.06**
  1. **Container**
     + **Std::vector**
       - **Functions of std::vector**
* **2023.11.08**
  1. **Container**
     + **Std::vector**
       - **Functions of std::vector**
  2. **Algorithms**
     + **Std::remove**
     + **Std::remove\_if**
     + **Std::unique**
     + **Std::erase**
     + **Std::erase\_if**
     + **Erase-remove idiom**
     + **Std::sort**
     + **Std::transform**
  3. **Std::ostream\_iterator**
* **2023.11.10**
  1. **Container**
     + **Std::deque**
  2. **Algorithms of sorting**
     + **Std::sort**
     + **Std::partial\_sort**
     + **Std::stable\_sort**
     + **Std::nth\_element**
     + **Std::partition**
* **2023.11.13**
  1. **Algorithms**
     + **Std::minmax\_element**
     + **Std::partition\_point**
     + **Std::is\_sorted**
     + **Std::is\_sorted\_until**
  2. **Heap Data Structer**
  3. **Std::queue**
  4. **Std::list**
  5. **Std::lower\_bound**
  6. **Std::upper\_bound**
  7. **Std::equal\_range**
* **2023.11.15**
  1. **Std::forward\_list**
  2. **Container Adapters**
     + **Stack**
     + **Queue**
     + **Priority\_queue**
  3. **Associative Container**
     + **Set**
* **2023.11.17**
  1. **Associative Container**
     + **Std::set**
     + **Std::map**
     + **Std::unordered\_set**
* **2023.11.20**
  1. **Associative Container**
     + **Std::unordered\_set**
  2. **Reference Wrapper**
  3. **Algorithms**
     + **Std::generate**
  4. **Function Adaptor**
     + **Std::bind**
* **2023.11.22**
  1. **Function Adaptor**
     + **Std::bind**
     + **Std::function**
     + **Std::mem\_fn**
     + **Std::not\_fn**
  2. **Std::array**
* **2023.11.24**
  1. **Std::tuple**
  2. **Structed binding**
  3. **Std::tie**
  4. **Std::apply**
  5. **Std::invoke**
  6. **Member Function Pointers**
* **2023.11.29**
  1. **Member Function Pointers**
  2. **Data Member Pointers**
  3. **Std::bitset**
  4. **Dinamik Ömürlü Nesneler**
* **2023.12.01**
  1. **Dinamik Ömürlü Nesneler**
  2. **Smart Pointer**
     + **Unique Pointer**
* **2023.12.04**
  1. **Smart Pointer**
     + **Unique Pointer**
     + **Shared Pointer**
* **2023.12.06**
  1. **Smart Pointer**
     + **Weak Pointer**
  2. **Curiously Recurring Template Pattern**
* **2023.12.08**
  1. **Input Output Operations**
* **2023.12.11**
  1. **Format**
     + **On off flag**
     + **Area flag**
  2. **Std::istreamstream**
  3. **İosstate**
* **2023.12.13**
  1. **std::istream\_iterator**
  2. **file operations**
* **2023.12.15**
  1. **File Operations**
     + **File positions pointer**
     + **gcount()**
* **2023.12.20**
  1. **Tie**
  2. **Lambda expression**
  3. **Std::ratio**
  4. **Std::chrono**
     + **duration**
* **2023.12.22**
  1. **Std::chrono**
     + **Duration**
     + **Duration\_cast**
     + **System\_clock**
  2. **UDL (user-defined literals)**
* **2023.12.25**
  1. **Std::chrono**
     + **System\_clock**
  2. **Vocabulary Type**
     + **Std::optional**
       - **Value\_or()**
       - **in\_place()**
     + **std::variant**
       - **hold\_alternative**
       - **get<>**
       - **in\_place\_index and in\_place\_type**
       - **monostate**
       - **get\_if**
* **2023.12.27**
  1. **Vocabulary Type**
     + **Std::variant**
     + **Std::any**
  2. **Std::visit**
  3. **Multi-Lambda**
* **2023.12.29**
  1. **Std::any**
  2. **Std::random**
* **2024.01.03**
  1. **Std::random**
  2. **Concurrency**
     + **Std::thread**
* **2024.01.05**
  1. **Std::thread**
     + **Get\_id**
     + **Sleep\_for and sleep until**
     + **Hardware\_concurrency**
     + **Thread exceptions**
  2. **Data Race**
* **2024.01.08**
  1. **Std::thread**
  2. **Mutex Class**
  3. **Mutex Wrapper**
* **2024.01.10**
  1. **Mutex**
  2. **Std::future and std::promise**
  3. **Std::async**