Got it. Here’s a **Session Plan (32 hours)** aligned strictly with **Stroustrup, *The C++ Programming Language* (4th ed., C++11)** — no C++17/20 features.

# Session Plan (32 hours) — C++11 per Stroustrup (4e)

**Format:** 16 sessions × **2 hours** each

1. **Tooling & Compilation Model** *(2 h)*  
   Topics: MSYS2 + VS Code verify; compile → link → run; translation units; headers vs sources; baseline warnings (-std=c++11 -O2 -g -Wall -Wextra -pedantic); “Hello, world”; stream I/O basics.
2. **Types, Declarations & Initialization** *(2 h)*  
   Topics: fundamental/compound types; declarations; **uniform initialization**; auto, decltype; nullptr; enum class; scope and using.
3. **Functions & Lambdas** *(2 h)*  
   Topics: overloading, default args, inlining; pass-by-value/reference; const member functions; function pointers; std::function; **lambda** syntax and captures.
4. **Classes I: Fundamentals & Copy Control** *(2 h)*  
   Topics: class/struct layout; invariants; constructors/destructor; **rule of 0/3**; explicit; =default, =delete; friends; static members.
5. **Move Semantics & RAII** *(2 h)*  
   Topics: rvalue references; **move ctor/assign**; std::move/std::forward; swap; resource ownership; RAII patterns (files, handles).
6. **Memory & Smart Pointers** *(2 h)*  
   Topics: new/delete (motivation), **std::unique\_ptr**, std::shared\_ptr, std::weak\_ptr; custom deleters; ownership diagrams; exception safety.
7. **STL Containers I — Sequences** *(2 h)*  
   Topics: std::array, std::vector, std::string, std::deque, std::list; iterator basics; iteration patterns; complexity & locality.
8. **STL Containers II — Associative/Unordered** *(2 h)*  
   Topics: std::map/set, std::multimap/multiset; **std::unordered\_map/set**; custom comparator/hash; iterator usage.
9. **Algorithms I — Core Patterns** *(2 h)*  
   Topics: non-modifying (e.g., find, count), modifying (copy, remove), sorting (sort, stable\_sort), numeric (accumulate); functors vs lambdas; erase–remove idiom.
10. **Algorithms II — Iterators & Idioms** *(2 h)*  
    Topics: iterator categories; partitioning and selection; transform; iota; composing algorithm pipelines (C++11 techniques); performance notes.
11. **I/O Streams & Formatting** *(2 h)*  
    Topics: iostream model; state bits/exceptions; iomanip formatting; file I/O (fstream, binary vs text); stringstream; locales (overview).
12. **Templates I — Fundamentals** *(2 h)*  
    Topics: function/class templates; template parameters (type & non-type); deduction; alias templates; **type traits** overview; constexpr functions (C++11).
13. **Templates II — Specialization & Techniques** *(2 h)*  
    Topics: full/partial specialization; templates + inheritance; CRTP mention; **SFINAE** basics with std::enable\_if; compile-time computations (intro).
14. **Error Handling** *(2 h)*  
    Topics: exceptions (throw/try/catch); noexcept; exception safety levels (basic/strong/nothrow); assertions; documenting error contracts.
15. **Standard Libraries: Utilities** *(2 h)*  
    Topics: **<chrono>** (durations, time points), **<random>** (engines, distributions), **<regex>** (basics); std::pair/tuple; std::bitset; <functional> (bind, ref).
16. **Concurrency (C++11) & Consolidation** *(2 h)*  
    Topics: **std::thread**, **mutex**, lock\_guard/unique\_lock, **condition\_variable**; futures/async/promises; atomics (overview); brief code review & integration of course idioms.

**Total:** 16 sessions × 2 hours = **32 hours**.

Notes:  
• All examples and practices target **C++11** as presented in the textbook.  
• Prefer STL containers/algorithms, RAII, move semantics, and standard C++11 concurrency; avoid later standards’ features (e.g., ranges, concepts, filesystem, format).