constexpr: evaluated at compile time if possible

consteval: always evaluated at compile time

constinit: values are initialized at program startup but are still mutable

inline member function: when called, its body is copied directly instead of making a member function call (can be marked inline by defining a member function in a class .hpp or by using the *inline* keyword in a class .cpp)

friend functions (prepended by keyword *friend*): not an actual part of their class, but can access private members (must be passed an instance of the class)

reinterpret_cast⟨T*⟩(oldPtr): does not change the underlying bits; only changes the type interpretation

explicit: prevents a constructor or conversion operator from being used for implicit type conversions; forces you to use explicitly cast or use constructor

- ⟨file⟩.good(); ⟨file⟩.fail(); ⟨file⟩.eof(); ⟨file⟩.bad()
- use after free; double delete; dangling pointer

when do you have to use member initializer list? when a member is const or a reference

- references must be initialized when declared and cannot be reassigned

g++ -Wall -Wextra -Werror -std=c++20 -o output.o *.cpp

Command-line redirection sends a program's input or output to/from files (e.g., ls ⟩out.txt), while piping sends the output of one program directly as input to another (e.g., ls | grep txt).

stack:

- faster and smaller than the heap
- stack pointer moves down when variables are initialized; when the pointer moves back up, all data below it falls out of scope

heap:

- slower and bigger than the heap, but allows dynamic memory allocation
- there is no heap pointer; compiler chooses where to store data and it is referred to by stack pointers
- memory must be manually managed (new, delete)

big five:

- copy constructor
  - Class(const Class& other)
  - perform deep copies
  - no delete because constructor
- copy assignment
  - Class& operator=(const Class& other)
  - if addresses are the same, return *this
  - delete dynamically allocated memory
  - perform deep copies
  - return *this
- move constructor
  - Class(Class&& other) noexcept
  - example of pass-by-rvalue-reference
  - perform shallow copies
  - set other object to null state
- move assignment
  - Class& operator=(Class&& other) noexcept
  - if addresses are the same, return *this
  - delete dynamically allocated memory
  - perform shallow copies
  - set other object to null state
  - return *this
- destructor
  - ˜Class()
  - delete dynamically allocated memory
  - set members to null state

value types:

- rvalue (right value)
  - prvalue (pure right value)
    - pure temporary with no ”identity”; no address and cannot be assigned to
    - e.g., literals, arithmetic
  - xvalue (expiring value)
    - named thing that is about to die
    - std::move(x) converts x to an xvalue
- glvalue (generalized lvalue)
  - lvalue (locator value)
    - named thing; persists; has an address; can be assigned to
    - run of the mill variable
  - xvalue (expiring value)
    - named thing that is about to die
    - std::move(x) converts x to an xvalue