Inheritance in C++

Michael Brodskiy
Professor: S. Shazli

March 29, 2023

- Inheritance
  - The mechanism by which one class can acquire the properties of another class, and then extend that class
  - We will utilize the "is a" relationship to define inheritance
    * For example, a car is a vehicle
- Hierarchy
  - Concepts at higher levels are more general
  - Concepts at lower levels are more specific (inherit properties of concepts at higher levels)
  - Derived classes are special cases of base classes
  - A derived class can also serve as a base class for new classes
  - There is no limit on the depth of inheritance allowed in C++ (as far as it is within the limits of the compiler)
  - Derived classes can inherit from more than one base class
- Three Benefits of Inheritance
  1. You can reuse the methods and data of the existing class
  2. You can extend the existing class by adding new data and new methods
  3. You can modify the existing class by overloading its methods with your own implementations
- Protected and Private Inheritance
  - With protected inheritance, public and protected members of Y become protected in X (*i.e.* classes derived from X inherit the public members of Y as protected)
  - With private inheritance, public and protected members of Y become private in X (*i.e.* classes derived from X inherit the public members of Y as private)
  - The default inheritance is private
- Virtual Functions
  - C++ uses virtual functions to implement run-time binding
  - To force the compiler to generate code that guarantees dynamic binding, the word virtual should appear before the function declaration in the definition of the base class
- `mmap()` and `munmap()`

  - Used to allocate memory space by mapping to new address
  - `mmap()` does the mapping
  - `munmap()` does the opposite — clears a memory address

- Certain devices can be mapped from physical to virtual space through memory mapping