* Any time you add or subtract floats of widely varying magnitudes you need to watch for loss of precision.
  + Store game time in a double, but it is okay to cast time deltas (which are always small numbers) from a double to a float and use the float to update the animation, physics, etc., if those systems’ interfaces take a float time value (for example, Box2D uses floats).
  + The limited precision of a float is only a problem if you are storing a very large number, such as game time elapsed after the program has been running for multiple days.
* RAII: Resource Acquisition Is Initialization
* Error handling
  + Develop a strategy early.
  + An invariant is logical condition for the members of an object that a constructor must establish for the public member functions to assume.
  + Have a constructor establish an invariant and throw if it cannot.
  + Throw exception types that you define yourself so there is no clash with other people’s code (for example, don’t throw a plain integer).
  + Destructors, deallocations functions like delete, and swap should never throw.
* B is a subclass of A
  + Unless B's ctor explicitly calls *one* of A's ctors, the default ctor from A will be called automatically *before* B's ctor body (the idea being that A needs to be initialized before B gets created).
  + *After* it exits, B's destructor will automatically call A's destructor.
* Prefer {}uniform initialization over =assignment or ()
  + TimeKeeper time\_keeper{Timer{}}; // unambiguous
  + TimeKeeper time\_keeper(Timer()); // ambiguous.. “most vexing parse” bug!
* size\_t is an unsigned integer that can express the size of any memory range supported on the our machine
* Single parameter constructors can be declared “explicit” to prevent being used for implicit conversion.