Move semantics provide a way for the contents of objects to be '**moved**' between objects, rather than copied, thus significantly changing the way we design code C++ by allowing things like return by value to be used a lot more often.

Polymorphism means "many forms", and it occurs when we have many classes that are related to each other by inheritance.

A pure Abstract class has only abstract member functions and no data or concrete member functions. In general, a pure abstract class is used to define an interface and is intended to be inherited by concrete classes. It's a way of forcing a contract between the class designer and the users of that class. The users of this class must declare a matching member function for the class to compile.

The override keyword serves two purposes: It shows the reader of the code that "this is a virtual method, that is overriding a virtual method of the base class." The compiler also knows that it's an override, so it can "check" that you are not altering/adding new methods that you think are overrides.

The inline functions are a C++ enhancement feature to increase the execution time of a program. Functions can be instructed to compiler to make them inline so that compiler can replace those function definition wherever those are being called. Compiler replaces the definition of inline functions at compile time instead of referring function definition at runtime.

The explicit function specifier controls unwanted implicit type conversions. It can only be used in declarations of constructors within a class declaration. For example, except for the default constructor, the constructors in the following class are conversion constructors.