Java

- Inner class cannot be overridden.
- Name hiding in Java only happens for static method, while it also happens for instance methods in C++.
- Arrays can be up-cast but not down-cast.
- Class with different type parameter is not compatible.
- Method with wildcard type parameter as type of argument cannot be called.
- `System.out.println(true ? Integer.valueOf(1) : Double.valueOf(2));` prints `1.0` because of static binding of overload.
  - Method overriding is determined with compile time type information.
  - Tertiary operator performs unboxing if both alternative are eligible.
- Generic array cannot be created, unless `<?>` which means nothing.
- Autoboxing and intern:
  - `-128 == new Integer(-128)` is `true` due to unboxing and arithmetic equality.
  - `new Integer(-128) == new Integer(-128)` is `false`.
  - `Integer.valueOf(-128) == Integer.valueOf(-128)` is `true` due to guaranteed caching of `-128` to `127`.
  - `Integer.valueOf(128) == Integer.valueOf(128)` is undefined.
  - Caching behavior of `Long.valueOf()` is also undefined.
- Calling `System.exit(0)` in `try` block can prevent `finally`

block from execution.

- Return value in `try` block can be overridden by another `return` in `finally` block, or the target object of the returned reference can be modified.