2. static vs dynamic typing. duck typed?

Clojure, like Lisp, is dynamically typed[1]. Lisp was a pioneer in adopting dynamic typing Paul Graham. There are no mutable states in Clojure ((Fogus & Houser, 2014)), so the following example is a work-around to illustrate dynamic typing.

```
usr=> (defn foo [x y] (/ x y)) ;; x / y
usr=> (foo \a 1) ;; char 'a' and int 1
ClassCastException java.lang.Character cannot be
cast to java.lang.Number  clojure.lang.Numbers.
divide (Numbers.java:159)
;; So the exception came from divide operator,
;; not the type checking of foo()
;; foo() does not require args to be certain types
;; so clojure is ducked typed as well.
```

As shown in the code, Clojure is **duck typed** as well.

[1] https://www.quora.com/Is-Scala-faster-than-Clojure. Michael Klishin seems to be an experienced developer of Clojure.

Fogus, M., & Houser, C. (2014). *The joy of clojure*. Manning Publ.