Development manifesto

A few words should be said regarding choices made during the development of this calculator. Split by section respective to each element.

Multiple handlers

A handler structure that uses a closure to generate appropriate handlers seemed like the logical choice to separate specifics of math functions from the operation of the handler, as well as for a well compartmentalized and generalized system that's easily expandable to more functions. One handler per function allows any requests made to other paths to naturally miss.

sync.Map

sync.Map is a type that needs a very specific set of circumstances to be faster than a normal map with manual locking. It also is unfortunately significantly less type safe than a normal map. However, the use in this server is close to the optimal usecase for a sync.Map . By storing pointers to answer structs we keep the contents of the map relatively static, with the number of reads of the map theoretically significantly outpacing the number of writes for common queries. Entries for keys are almost never written to twice. It'd potentially be worth stress testing both to see which wins out on speed, and going with normal maps and mutexes if it's close.

Errors

A decision was made to not store errors in cache. The main purpose of the cache is to speed up returns on requests for common queries and in this particular case errors are based on checking of the inputs, which happens very soon after receiving the request. Thus, caching the errors does not make a big difference in the speed with which they are returned, but does grow the size of the cache, which can have negative effects on the performance of the cache. If commands were later added that require substantial processing before erroring out, errors could be readded to the program.