1. **Hashing strings up to 32 characters**

I would have an m value of 2^32 so to cover all characters. With a p being the length of the specific string. Then summing each character would be string s = s[p]*p^(1)+s[p-1]*p^(2)…..s[1]*p^p this mod m. I believe this would give a value of very low probability of collision and would be decently complex. Max length would be m.

Hashing a string of 1000 characters

I would take them in in portions of 8. These portions would take the summation of each character in decimal form. That summation mod an 8 digit prime number. The results of each of these summations would be summed to be then modded by that same 8 digit prime number.

Hashing a floating point num with 8 sig figs

a simple method would be to take to take the floor of 4 of the numbers in the form (xxxx.yyyy). If there are values non-zero in the last 4 take the floor of those values as if they were a separate integer and add it to the result of the first operation giving your the final hashed value. floor(xxxx) + floor(yyyy) = hashed integer.

hashed function for a hexidecimal address memory:

each memory location for the system is going to be unique so sorting those values could be done with just the value. There will be non repeats. Each value will be in the form 0xffffffffffffffff and can be hashed to be the binary representation of that hex location or use the hex given for the location. a location could be 0x7fff5fbff86c and that is how it would be stored as each location would be unique with the one following it being 0x7fff5fbff86d