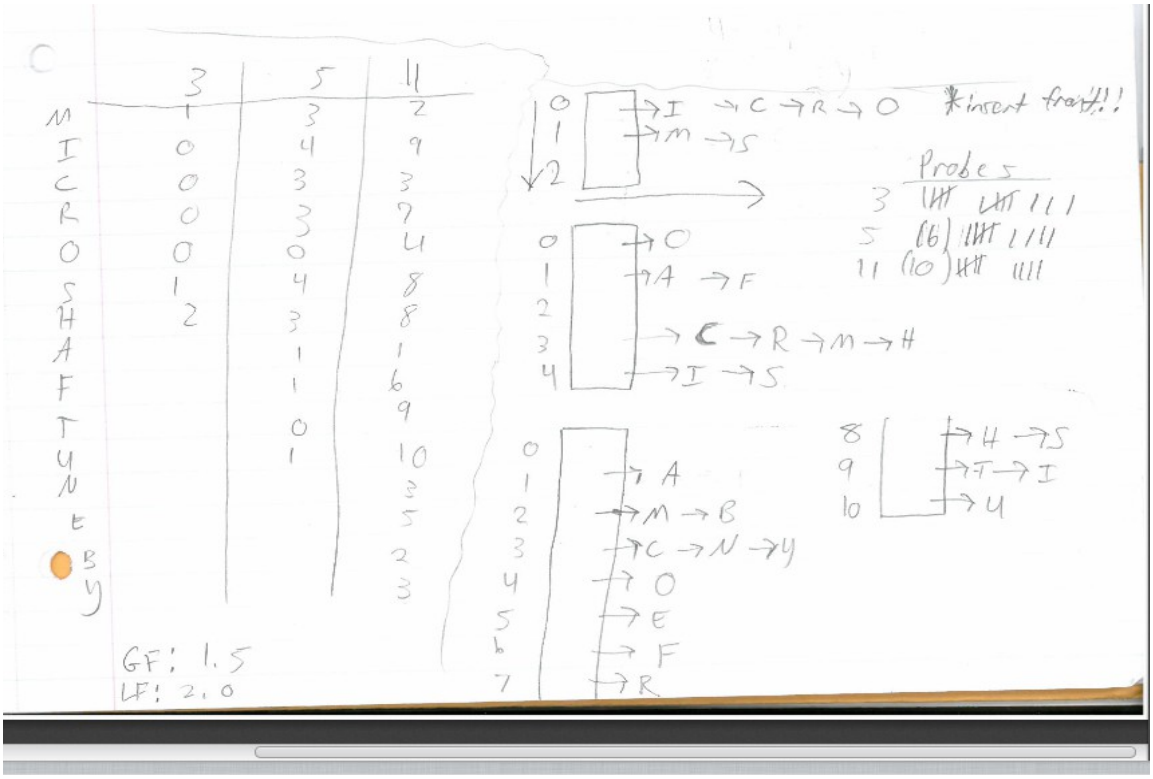
CS280 – HashMaps
March 7, 2016

Collision Resolution by Chaining

http://azrael.digipen.edu/~mmead/www/Courses/CS280/Hashing-1.html

- Average case (w/ good hash function) lists are all similar size
- Hash table contains pointers to data, not the data itself
- Keep in mind, we will **always** have collisions, it is how we handle them
- We push to the front of the LL (diagrams show back, but we are pushing to front)
- Although sorted data might be a little faster, lists will be so small that we don't have to worry about it
- Splaying – Move recently accessed items to front of list
- Do NOT have to check for duplicates when re-inserting, can't be duplicates
- Re-use nodes instead of freeing and re-allocating
- Will lose points if two separate insert functions (just us a flag)



Inserting into a hash table with chaining

- Clear deletes the nodes but not the table itself
- No need to use the STL for this assignment

Hash Functions

- Point of a good hash function is to make even smal differences in input produce large changes in output
- Hash functions are also designed to be very fast
    - Division and modulus in loops are bad