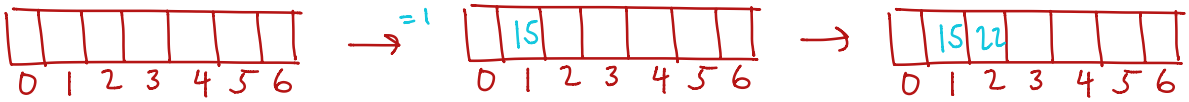
## Question 2

A) $h(K) = K \% m$

K = key
m = table size

Insert 15

$h(15) = 15 \% 7$
$= 1$




Insert 22

$h(22) = 22 \% 7$
$= 1$ collision
$h(22) = (22 + 1^2) \% 7$
$= 2$

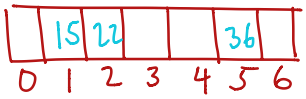



Insert 36

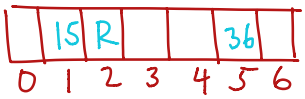
$h(36) = 36 \% 7$
$= 1$ collision

$h(36) = (36 + 1^2) \% 7$
$= 37 \% 7$
$= 2$ collision

$h(36) = (36 + 2^2) \% 7$
$= 40 \% 7$
$= 5$




Delete 22




Find 36

$j = 0$
ht[i]
where $i = (36 + j^2) \% 7$
$= 1$

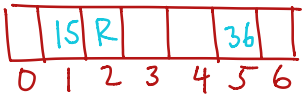
↳ if ht[i] == empty → not found
else if ht[i] == 36 → found
else j++;

JD... find 36

$j = 0$
$h(36) = 36 + j^2 \% 7$
↳ $(36 + 0^2) \% 7$
$= 1$
@ index 1 we have 15 so move on → j++
$h(36) = (36 + 1^2) \% 7$
$= 37 \% 7$
$= 2$
@ index 2 we do not have 36 so we move on

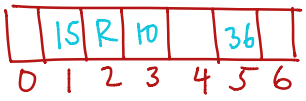
$h(36) = (36 + 2^2) \% 7$
$= 5$
@ index 5 we have 36 so

Found @ index 5



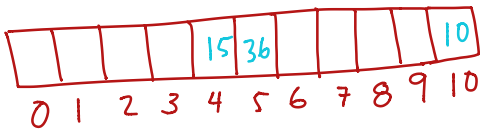

Insert 10

$h = 10 \% 7$
↳ 3




Load factor greater than 1/2 → Resize to size 11!

$h(k) = k \,\%\, 11$ — 22 is not there since it was removed




$h(15) = 15 \,\%\, 11 = 4$

$h(22) = 22 \,\%\, 11 = 0$ → Removed though

$h(36) = 36 \,\%\, 11 = 5$

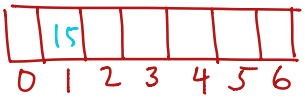
$h(10) = 10 \,\%\, 11 = 10$

---

Part B

insert 15

$h(k) = 15 \,\%\, 7 = 1$

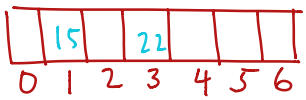



insert 22

$h(22) = 22 \,\%\, 7 = 1$ collision

$h_2(22) = 3 - (22 \,\%\, 3) = 2$

$(1+2) \,\%\, 7 = 3$

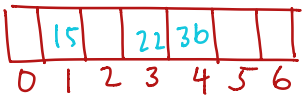



insert 36

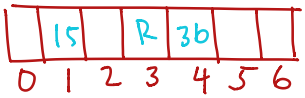
$h(36) = 36 \,\%\, 7 = 1$ collision

$h_2(36) = 3 - (36 \,\%\, 3) = 3$

$(3+1) \,\%\, 7 = 4$




Remove 22 →




Find 36

j = 0

ht[i] = 36

$i = h(36) = 36 \,\%\, 7 = 1$

ht[1] ≠ 36

↓ j++ ↓ $h_2$

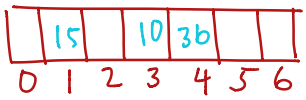
$i = ((36 \,\%\, 7) + (j * 3 - (36 \,\%\, 3))) \,\%\, 7$

$= (1 + 3) \,\%\, 7 = 4$

h[4] = 36 ✓ found 36 @ index 4

insert 10

$h(10) = 10 \,\%\, 7 = 3$



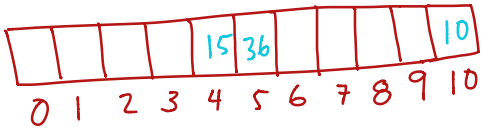

* since removed elements are considered part of load factor, the LF is greater than 1/2 so rerite to 11.

$h(15) = 15 \,\%\, 11 = 4$

$h(22) = 22 \,\%\, 11 = 0$ → Removed, however

$h(36) = 36 \,\%\, 11 = 5$

$h(10) = 10 \,\%\, 11 = 10$



Part C

The probability that hi(x) = p is 1/m, so the probability it returns p for all 3 filters is 1/m^3. So, that's the probability of returning true. The probability that it returns true and then that the index is actually false is 1/m^3 * m/3, so 1/3*m^2. Then, by linearity of expectations (LOE), you are expected to get 27 * 1/3*m^2. = 9/m^2 false positives

QUESTION 3 is on next page

Question 3 cacheLRU.h analysis

1. For testing purposes, the first file I used was the hamlet text that I found online. Since this was recommended in the instructions for a larger input file, I thought I would use it. The second text file I used was that of Midsummer Night's Dream by Shakespeare but instead of using the whole play, I decided to use only a portion of it, so about 12,013 words. For the 3rd file, I used a text file containing a list of randomly generated integers.
2. For the Hamlet text file, I decided to make the cache's capacity 5,000. For the Midsummer text file, I decided to make the cache's capacity 1,000 since it was a smaller text. For the numbers text file, I kept the capacity at 1,000.
3. For the Hamlet text file, the total number of rotations was 121,657 rotations (57,316 left rotations + 64,341 right rotations). For the Midsummer text file, the total number of rotations was 41788 (16,997 left rotations + 24,791 right rotations). For the numbers text file, the total number of rotations was 48,970 (18,834 left rotations + 30,136 right rotations).
4. The hamlet text file had a size of 32,000 words. The Midsummer text file had a size of 12,013 words. The numbers text file had a size of 12,000 numbers.
5. For the Hamlet file, 121,657 total rotations/ 32,000 words is about 3.8 average rotations per word inserted. For the Midsummer file, 41,788 total rotations/ 12,013 words is about 3.5 average rotations per word inserted. For the numbers text file, 48,970 total rotations/ 12,000 words is about 4.08 average rotations per number inserted.
6. For the hamlet text file, the total amount of items removed was about 5,900 items. For the Midsummer text file, the total amount of items removed was about 6656 items and I think this is due to the decrease in capacity compared to the larger Hamlet file. For the numbers text file, the total number of removals performed was 10,983 items. I think this removal was so high because there weren't many duplicates in the number text file.
7. For the two mid-sized text files, Midsummer and the number file, I see a difference in the number of removals performed in that Midsummer was significantly less coming in at 6,656 items removed whereas the number file had about 10,983 items removed. I think that the reason being is that since the Midsummer text had a lot of duplicate words, they were not added to the overall size of the cache, so the capacity was not reached as much. When a duplicate word is found, it is not added again. With the numbers text file, there were not as many duplicates therefore allowing the cache's capacity to be reached far more times, and thus we had to delete/remove a min leaf.
8. What I was surprised by running the two tests on the mid-sized files, Midsummer and number file, was that while the total number of items in Midsummer was greater than number file by only 13 items, the average number of rotations for midsummer was 3.5 rotations whereas for the numbers file it was 4.08 rotations on average.