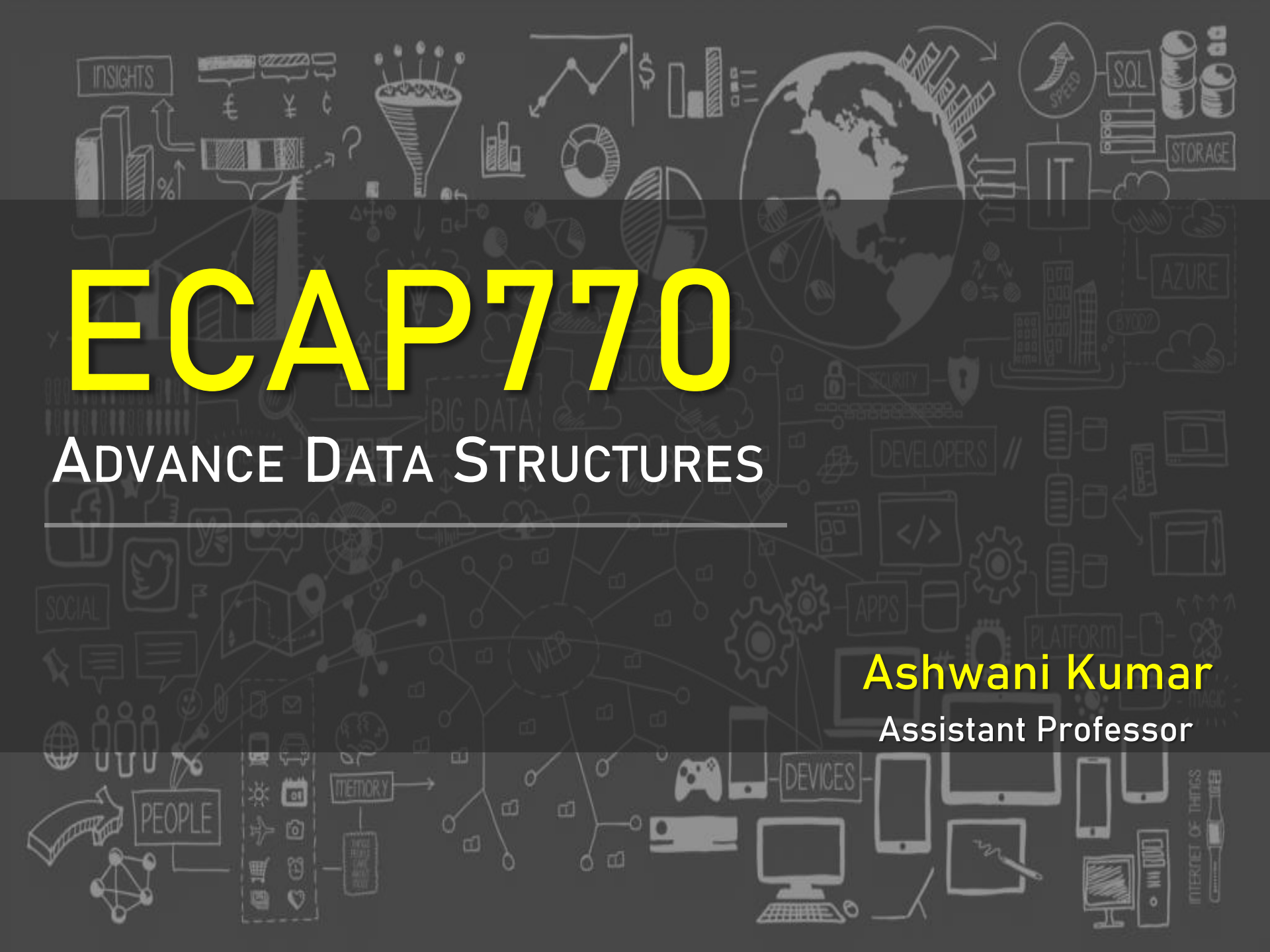


ECAP770

Advance Data Structures

Ashwani Kumar

Assistant Professor

Learning Outcomes

After this lecture, you will be able to

- digit folding method / pairing method,
- multiplication Method.

Hash function

- A hash function is any function that can be used to map a data set of an arbitrary size to a data set of a fixed size, which falls into the hash table.
- The values returned by a hash function are called hash values, hash codes, hash sums, or hashes

Characteristics of hash function

- Uniform Distribution : For distribution throughout the constructed table.
- Fast : The generation of hash should be very fast, and should not produce any considerable overhead.
- Less collisions : Collisions occur when pairs of elements are mapped to the same hash value. These should be avoided

Characteristics of hash function

- Uniform Distribution : For distribution throughout the constructed table.
- Fast : The generation of hash should be very fast, and should not produce any considerable overhead.
- Less collisions : Collisions occur when pairs of elements are mapped to the same hash value. These should be avoided

Characteristics of hash function

- Uniform Distribution : For distribution throughout the constructed table.
- Fast : The generation of hash should be very fast, and should not produce any considerable overhead.
- Less collisions : Collisions occur when pairs of elements are mapped to the same hash value. These should be avoided

Hash function

- Methods for calculating the hash function
  - Division method
  - Mid square method
  - Digit folding method / pairing method
  - Multiplication Method

Digit folding method

- The folding method for constructing hash functions begins by dividing the item into equal-size pieces (the last piece may not be of equal size).
- These pieces are then added together to give the resulting hash value.

Folding method

- Fold shift
- Fold boundary
  - The left and right numbers are folded on fixed boundary between them and the centre.
  - The two outside values are then reversed.

Digit folding method

- Case 1: if hash table size is 100 (0-99) and sum in 3 digits, then we will ignore last carry, if any.

Or

- Case 2: if hash table size is 100 (0-99) and sum in 3 digits, then we need to perform the extra step of dividing by 100(size of table) and keeping the remainder.

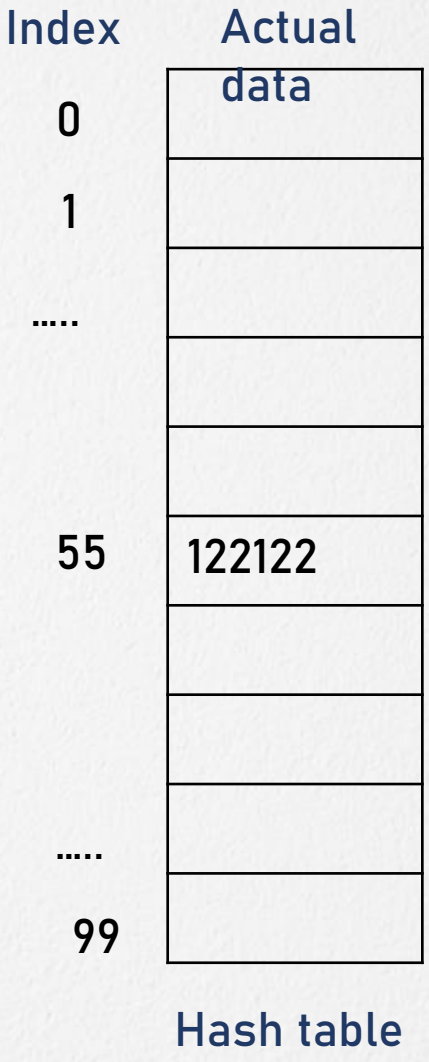
Folding method: case 1

Key: 122122

Parts: 12 21 22

Sum: 55

Hash value: 55



Folding method: case 2

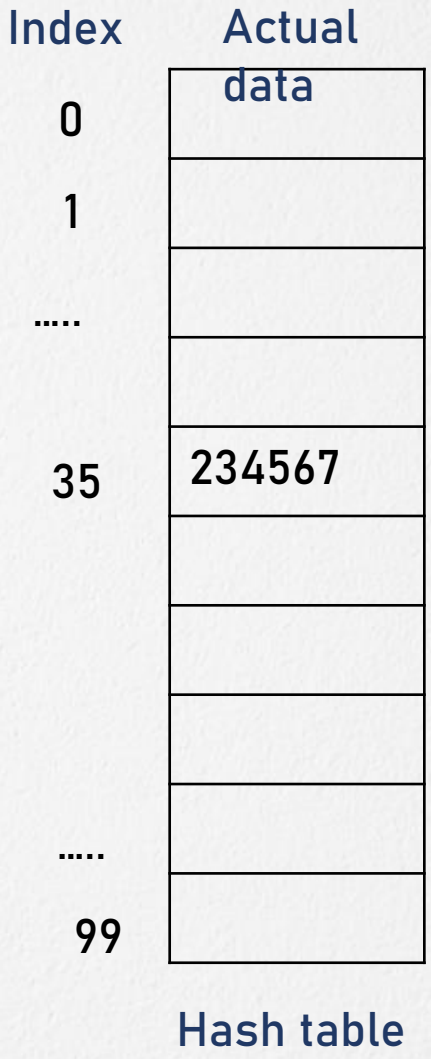
Key: 234567

Parts: 23 45 67

Sum: 135

Ignore last carry

Hash value: 35



Folding method: case 3

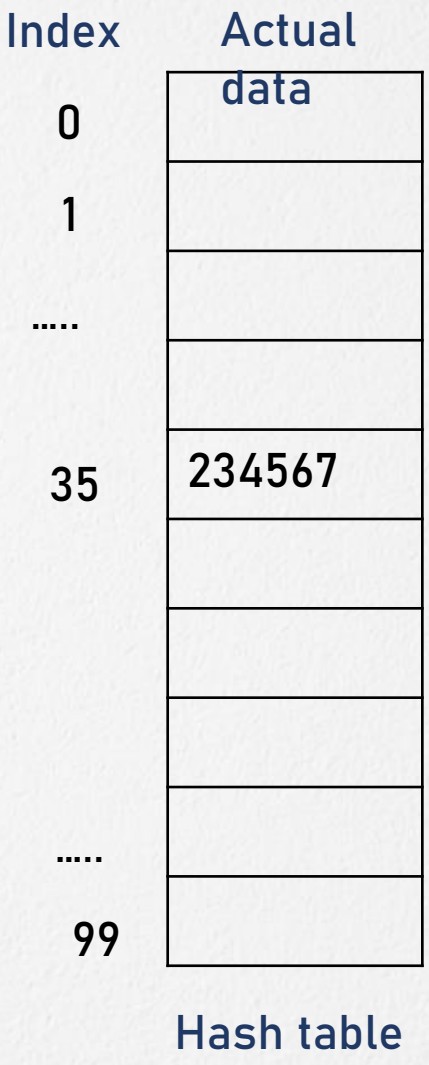
Key: 234567

Parts: 23 45 67

Sum: 135 mod 100 = 35

Sum mod with table size

Hash value: 35



Fold boundary: case 1

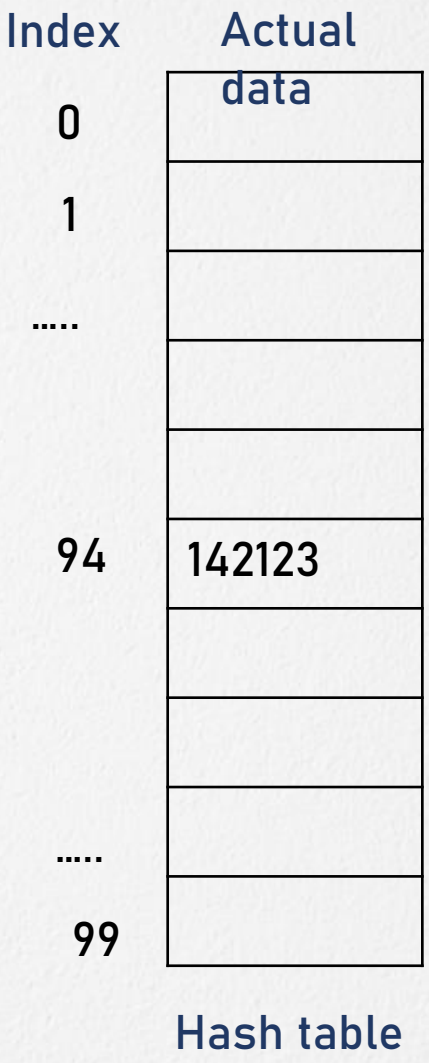
Key: 142123

Parts: 14 21 23

41 21 32

Sum: 94

Hash value: 94



Fold boundary: case 2

Key: 354027

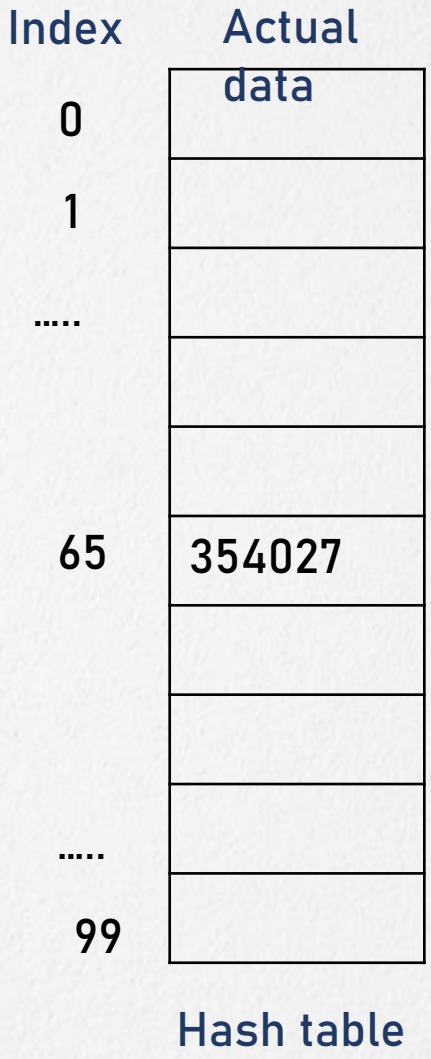
Parts: 35 40 27

53 40 72 value reversed

Sum: 165

Ignore last carry

Hash value: 65



A

Multiplication Method

h(key) = floor(table size *(key *A)) % size

Or

h(k) = floor(n(kA mod 1))

- K= key
- A is constant value between 0 and 1
- Knuth recommends A = 0.6180339887… (Golden Ratio)

Multiplication Method

1) Choose constant
2) Multiply key k by A
3) Extract fractional part of k A (this gives us a number between 0 and 1)
4) Multiply fractional part by m and take floor of the multiplication (this transforms a number between 0 and 1, to a discrete number between 0 and m-1 that we can map to slot in the hash table)

Example: Multiplication Method

k=34

Size of table = 100

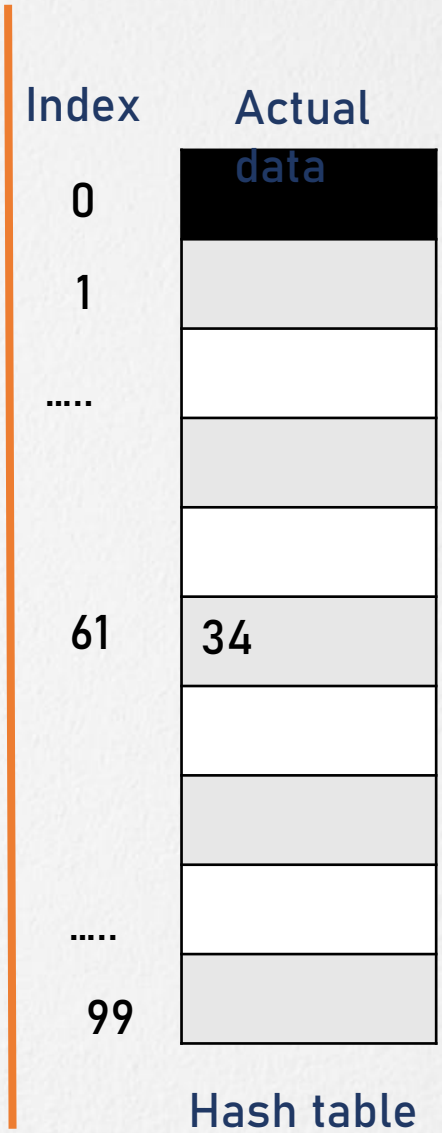
A=0.618033

h(34) = floor (100(34 * 0.618033)) % 100

=floor (3461.80) %100

=3461 %100

=61



That's all for now...