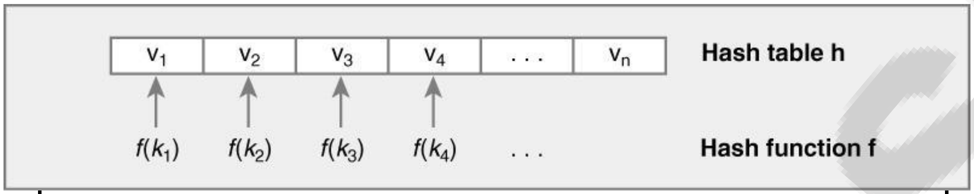
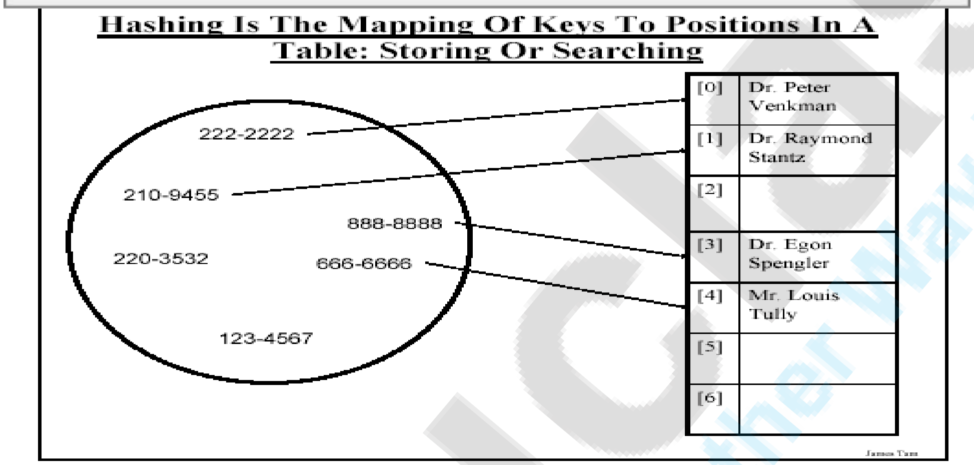
# Chapter 3 Hashing

- A function that transforms a key into a table index is called a hash function i.e. process to do key-to-address transformation
- H(key) ➔ address

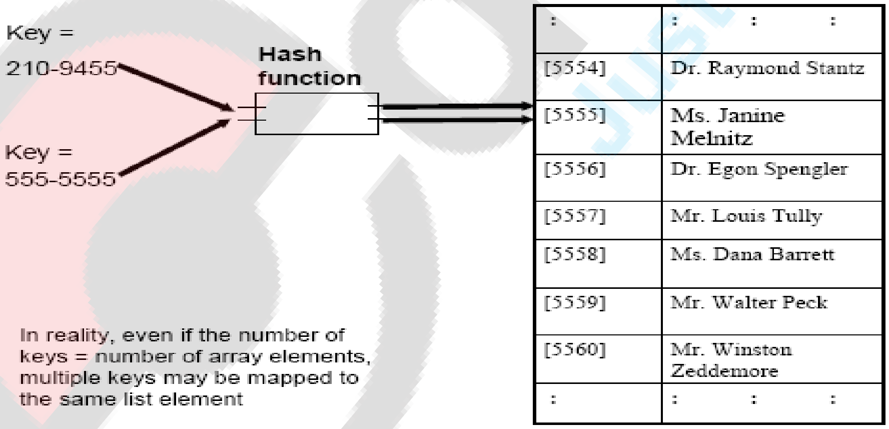



Hashing Is The Mapping Of Keys To Positions In A Table: Storing Or Searching
222-2222
210-9455
888-8888
220-3532
666-6666
123-4567
James Tam

| | |
|---|---|
| [0] | Dr. Peter Venkman |
| [1] | Dr. Raymond Stantz |
| [2] | |
| [3] | Dr. Egon Spengler |
| [4] | Mr. Louis Tully |
| [5] | |
| [6] | |

## Collision:

- A Collision occurs when hashing algorithm produces an address for an insertion key and that address is already occupied.
- k1 ≠ k2, h(k1) = h(k2)


Key = 210-9455
Key = 555-5555
Hash function
In reality, even if the number of keys = number of array elements, multiple keys may be mapped to the same list element

| | |
|---|---|
| : | : : : |
| [5554] | Dr. Raymond Stantz |
| [5555] | Ms. Janine Melnitz |
| [5556] | Dr. Egon Spengler |
| [5557] | Mr. Louis Tully |
| [5558] | Ms. Dana Barrett |
| [5559] | Mr. Walter Peck |
| [5560] | Mr. Winston Zeddemore |
| : | : : : |

## Terminology:

- **Synonyms**
    - Keys which hash to the same value.

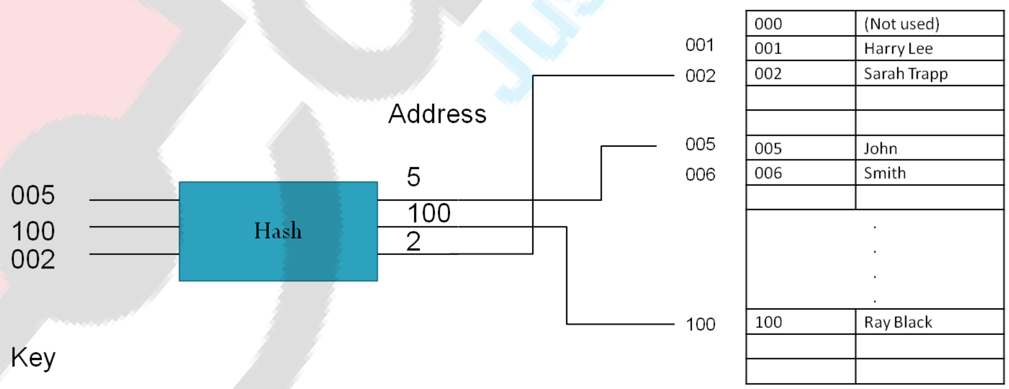
- **Home Address**
  - Address produced by the hashing algorithm
- **Prime Area**
  - Memory that contains all home addresses
- **Probe**
  - When there is need to locate an element in a hashed list, the algorithm used to insert the element must be applied first.
  - If desired element is not at calculated location then apply collision resolution algorithm to determine next location.
  - Continue until we find the element or determine that it is not in the list.
  - Each calculation of an address and test for success is called probe.

**Basic hashing techniques:**

1. Direct
2. Subtraction
3. Modulo-division
4. Digit extraction
5. Midsquare
6. Folding
7. Rotation
8. Pseudorandom generation

### 1. Direct Address Method:

- Key is the address without any manipulation.
- Applicable when we can afford to allocate an array with one position for every possible key.



| | |
|---|---|
| 000 | (Not used) |
| 001 | Harry Lee |
| 002 | Sarah Trapp |
| | |
| | |
| 005 | John |
| 006 | Smith |
| | |
| | .<br>.<br>.<br>. |
| 100 | Ray Black |
| | |
| | |

### 2. Subtraction:

Used for keys that are consecutive but do not start from 1.

E.g. 1000 to 1100. Subtract 1000 from all keys.

Addresses generated will be from 000 to 100.

3. **Modulo – Division Method:**

Key is divided by array size and uses remainder plus 1 for the address.

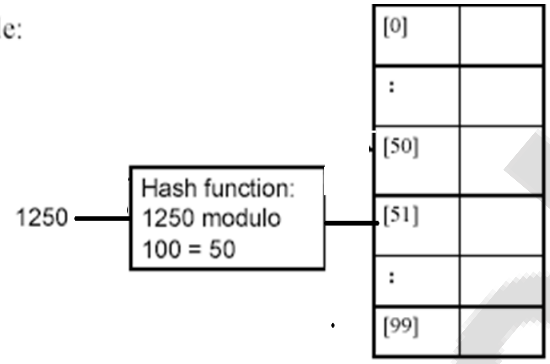
Address = key MODULO list size + 1

To reduce the number of collision.

list size should be prime.

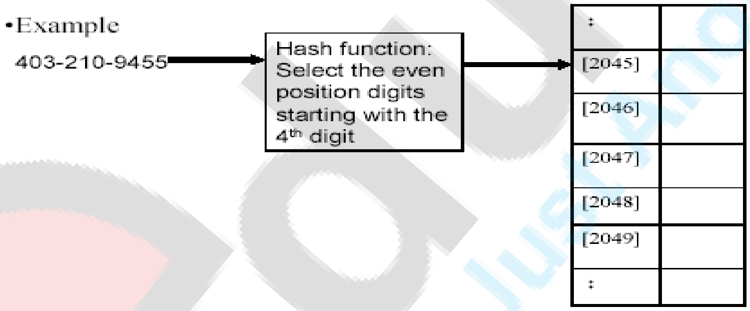
**Address = key MODULO listsize + 1**

•Example:



4. **Digit Extraction:**

Selected digits are extracted from the key and used as the address.



5. **Mid-square Method:**

Key is square and the address is selected from the middle of the square number.

Example:

Key = 9452

9452 * 9452 = 89340304 => 3403

**Disadvantage:** Say if key is 6 digit, square would be 12 digit, beyond the integer limit of most computers.

**Variation :** Apply method on say, first 3 digits of the number.

Example: 379452 => 379 * 379

6. **Folding Method:**

- There are two folding methods
  1. Fold shift
  2. Fold boundary
- **Fold Shift :** key is divided into number of parts say k1,k2,…..,kn where each parts has the same number of digits except the last part , which can have lesser digits.
  Add all these parts and ignore last carry.
  Example:
  Key=123456789

```
   123
 + 456
   789
  1368
```

Discard 1 **so the address is 368**

- **Fold Boundary:** left and right numbers are folded on a fixed boundary between them and the center number . This results in two outside values are being reversed .

Example:

Key = 123-456-789

```
       321
       456
   +   987
      1764
```

Discard 1 **so the address is : 764**

7. **<u>Rotation Method:</u>**

This method is used with other hashing methods.
This method rotate the last character to front.
Used when keys are identical except for last character.
This method used to minimized the effect of creating synonyms.

Example:

60010**1** → **1**60010
60010**2** → **2**60010
60010**3** → **3**60010

Now apply fold shift → 16 + 00 + 10 => 26
26 + 00 + 10 => 36
36 + 00 + 10 => 46

Spreading data more evenly across the address space

8. **<u>PseudoRandom Method:</u>**

Key used as seed in a pseudo random number generator.

Resulting random number scaled into possible address range using modulo division.

The common random number generator: Y = ax + c

x = key , a = coefficient , c = constant

Ex. X= 1212 a=10 c= 5

y = (10 * (1212) + 6)modulo 10 +1

where list size = 10

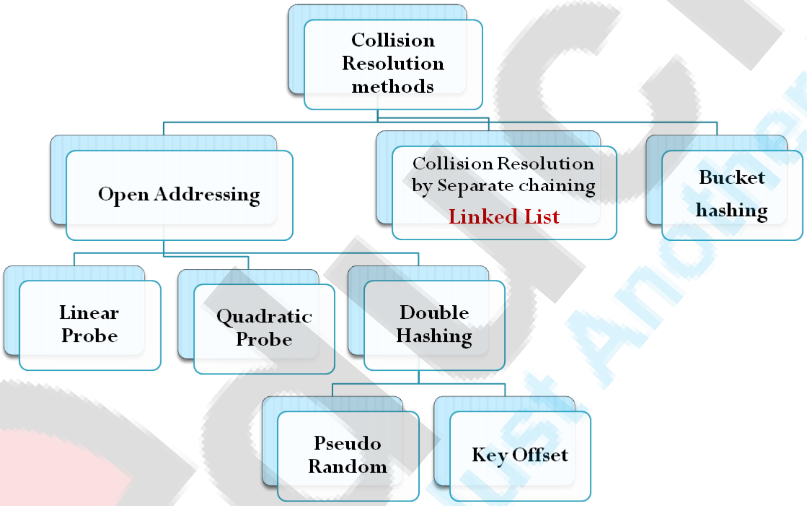
y = (12120+6)mod 10 +1

= 12126 mod 10 +1

= 7

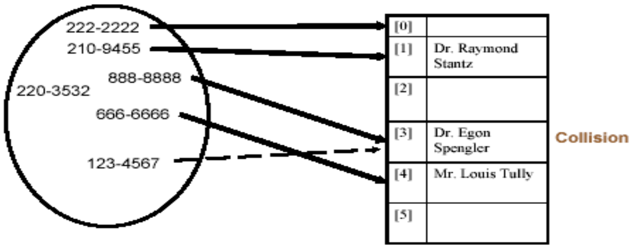
## Collision Resolution:

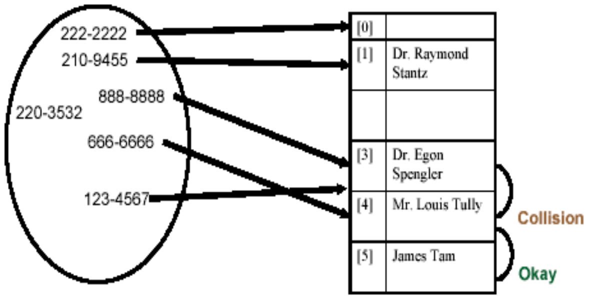
- A collision is a phenomenon that occurs when more than one key maps to same slot in the hash table.



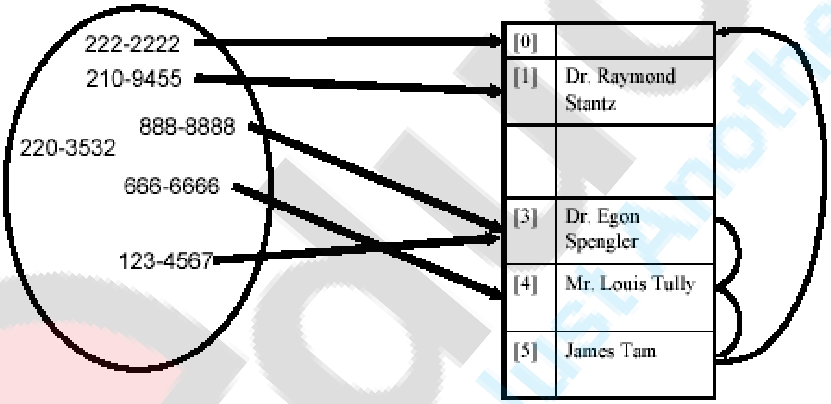
## Linear Probing:

When a collision occurs, sequentially search the table until an empty location is found.





- The table is treated as circular: When the end of the table has been probed, begin probing at the beginning.



**Linear Probing:**

- In linear probe ,when data cannot be store in to home address , we resolve the collision by adding one to the current address.
- It uses the hashing function
  h[(k,i)= [h'(k) + i ] for i= 1,2 ---- m
  Where m is size of hash table
  h'(k) = k mod m
  i is probe number

**Definition:**

- **synonyms** :Keys which hash to the same value.

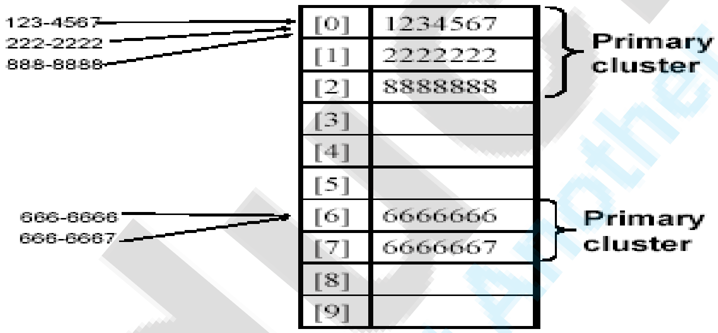
- **Collision:** An attempt to store a record at an address which does not have sufficient room .
- **packing density :**The ratio of used space to allocated space.
- **home address:** The address produced by the hashing of a record key.

**Advantages/ Disadvantages:**

**Strength:**

1. As long as there is an unused location in the table, this approach will find it (eventually) and is simple. Data remains near home address.
2. They are quite simple to implement
3. Data tends to remain near their home address.

**Weakness:** Table entries tend to cluster around parts of the table leaving some continuous sections that are occupied and others empty (uneven distribution) – Primary clustering



## Quadratic probing :

It uses the hash function

$h[(k,i)= [h'(k) + C_1 i + C_2 i^2 ]\text{mod } m$ for i=0 , 1,2 ---- m-1

Where

m size of hash table

$h'(k) = k \text{ mod } m$

i is probe number

$C_{1,}\ C_2 \neq 0$ are auxiliary constant

## Double hashing : Pseudorandom

- Rather than using key as a factor in random number calculation we use the address .
- Hashing function ;

y= (ax + c) modulo array size +1

as a= 3 and c= -1

y= ( 3 x 2 + (-1 )) modulo 307 +1
= 6

## Double hashing : key offset

- Key offset calculates the new address a function of the old address and key .
- New address can be calculated by adding the quotient of the key divided by array size to the address .
- Offset = key / array size
- Address= ((offset + old address )modulo array size )+1

Example:

- Key = 123456
- Array size = 23
- Address = 123456 %23 +1
  = 15 + 1 = 16

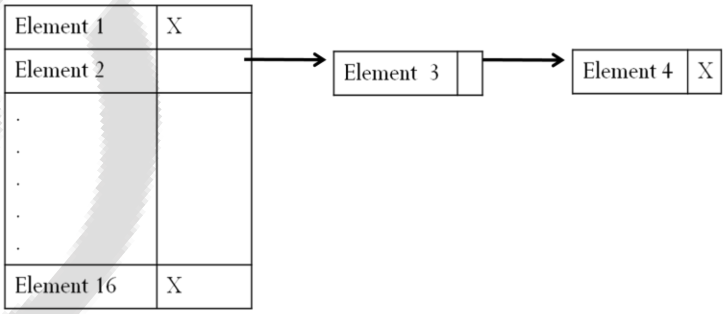
If 16th location is already occupied , it produces a collision , to resolve we have to find the new address :

Offset = 123456 / 23 = 5367

Address = (( 5367 + 16 ) mod 23 ) +1 = 1+1 = 2

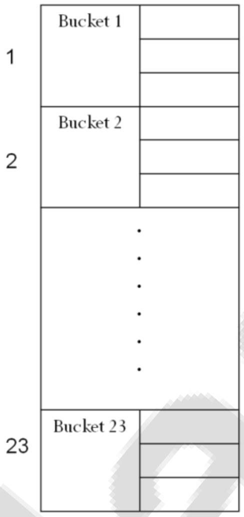
## Link list resolution:

- A link list is ordered collection of data in which each element contains the location of next element .
- Link list uses the separate area to store collision and chains all synonyms together in a link list .
- It uses two storage area , prime area and the overflow area .
- Each element in the prime area contains an additional field , a link header pointer to a link list of overflow data in the overflow area .



## Bucket hashing:

- Bucket accommodate multiple data occurrences , and due to this collision are postponed until bucket is full.
- From figure , each address is big enough to hold data about 3 student. Under this assumption , there would not be collision until we tried to add 4th student to an address.



**Disadvantage:**

- It use significantly more space because many of the bucket will be empty or partially empty at any given time.
- It does not completely resolved the collision problem

**Question:**

1. **Using modulo division and linear probing, store keys in an array of 19 elements. What are the number of collisions and density of list after all elements are stored?**
   **224562, 137457, 214562, 140145, 214576, 162145, 144467, 199645, 234534**

**Ans:**

224562 % 19 + 1 = 2

137457 % 19 + 1 = 11

214562 % 19 + 1 = 15

140145 % 19 + 1 = 2 =>3 collision

214576 % 19 + 1 = 10

162145 % 19 + 1 = 19

144467 % 19 + 1 = 11 =>12 collision

199645 % 19 + 1 = 13

234534 % 19 + 1 = 18

Density 9/19 = 47 %

2. **Do it using digit extraction (first, third, fifth) and quadratic probing.**

**224562, 137457, 214562, 140145, 214576, 162145, 144467, 199645, 234534**

**Answer:**

224562 => 246 % 19 + 1 = 19

137457 => 175 % 19 + 1 = 5

214562 => 246 % 19 + 1 = 19 collision

$19 + 1^2$ => 20 % 19 + 1 =>2

140145 => 104 % 19 + 1 = 10

214576 => 247 % 19 + 1 = 1

162145 => 124 % 19 + 1 = 11

144467 => 146 % 19 + 1 = 14

199645 => 194 % 19 + 1 = 5 collision

$5 + 1^2$ => 6 % 19 + 1 => 7

234534 => 243 % 19 + 1 = 16

**Q. What is hashing? Define the terms collision, probe and the load factor. Insert the keys 99 33 23 44 56 43 19, using the division method and quadratic probing as a collision resolution method into a list of size 10. also find the number of collisions, probes for each element and the density of the list.** (May 2010)

**Answer:**

- 99

  99%10+1 = 10

- 33

  33%10 +1 = 4

- 23

  23%10+1 = 4 collision, probe 1

  $4+1^2$ =5

  5 % 10 +1 = 6

- 44

  44%10+1 = 5

56%10+1 = 7

- 43

  43%10+1 = 4 collision, probe1

  $4 + 1^2 = 5$

  5 % 10+1 = 6, collision, probe 2

  $6 + 2^2 = 10$

  10 % 10 + 1 = 1

- 19

  19%10 + 1 = 9+1=10 collision, probe 1

  $10 + 1^2 = 11$

  11%10 + 1 = 2