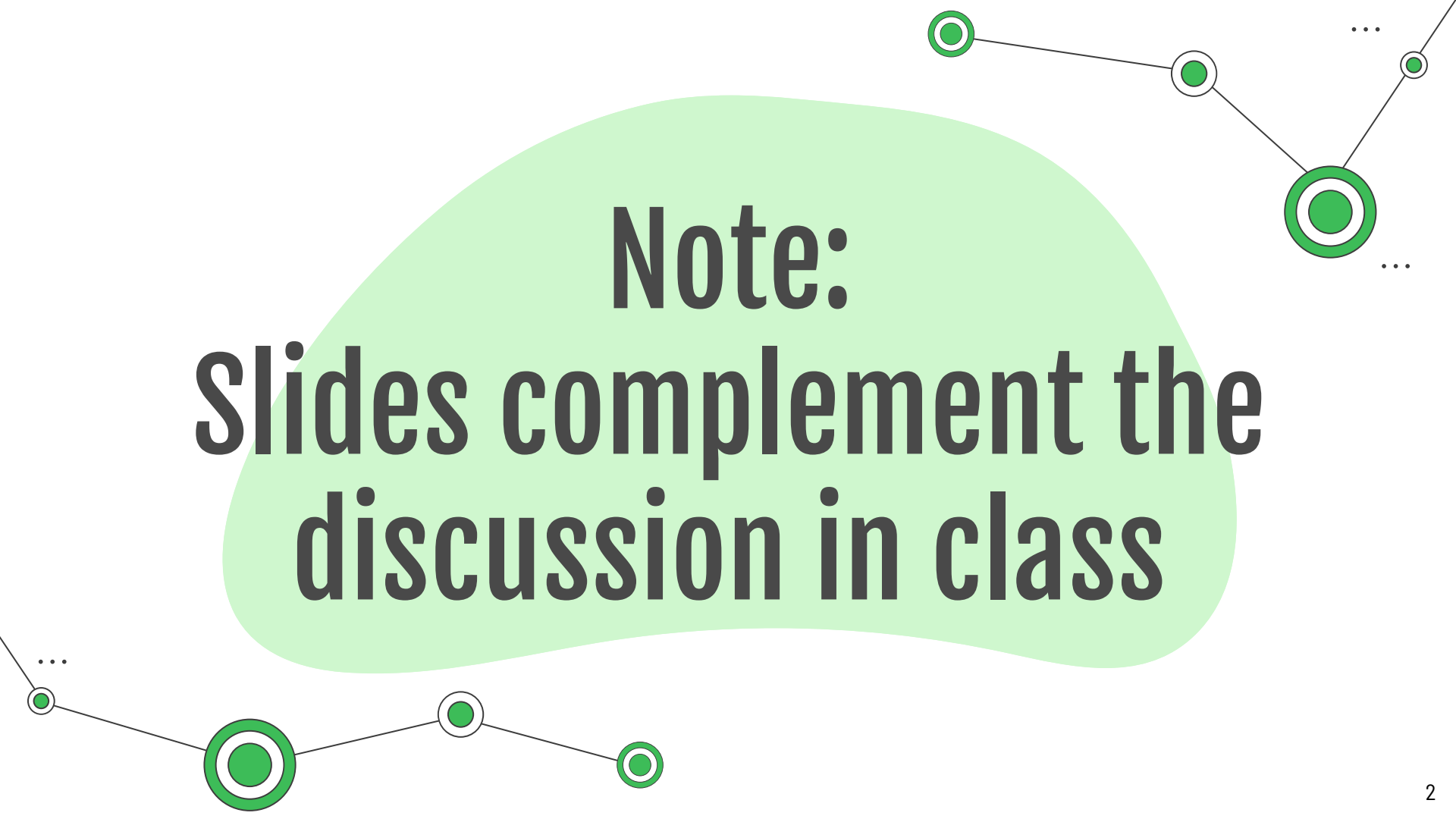


Hash Table with Chaining

CS 251 - Data Structures
and Algorithms

A decorative network diagram consisting of green circular nodes connected by thin black lines. The nodes are arranged in a non-linear fashion, with some having concentric circles. Ellipses (...) are used to indicate that the network continues beyond the visible nodes.

Note:
**Slides complement the
discussion in class**

Table of Contents

01

Hash Table

Hashing and tables working together

02

Chaining

Store items in Linked Lists



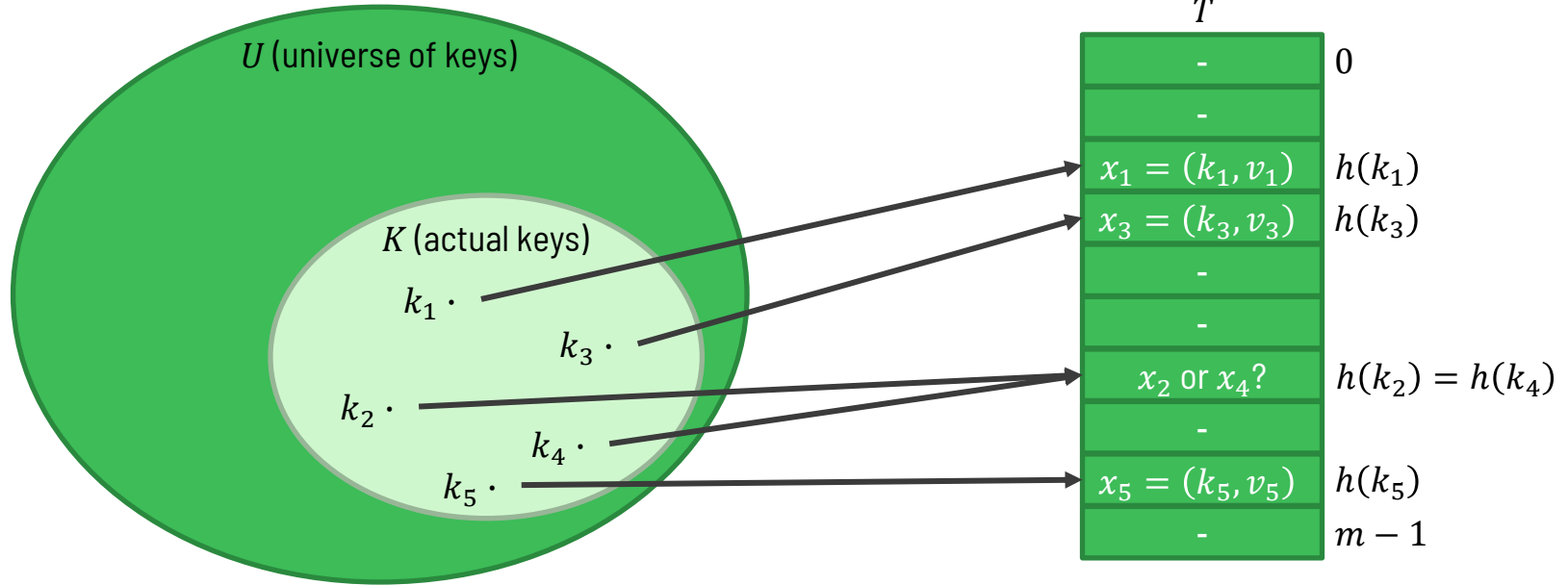


01

Hash Table

Hashing and tables working together

Hash Table



$$h: U \rightarrow \{0, 1, 2, \dots, m-1\}$$

Conventions



Unique keys

Each key k can only have one value.



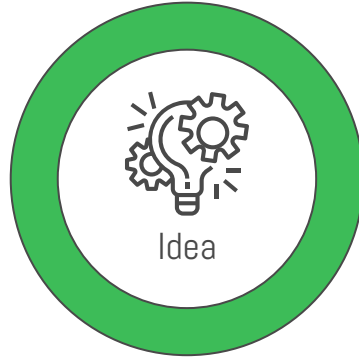
Replace

If entry (k, v) exists, and we try to insert (k, w) , then (k, v) is replaced with (k, w) .



ADT/API

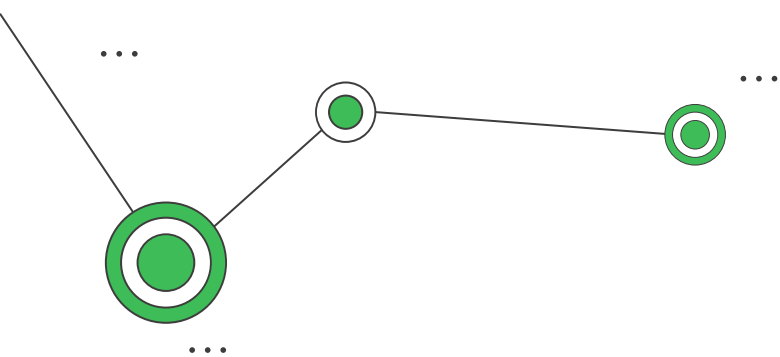
Put(key, value)
Get(key)
Delete(key)
Contains(key)
isEmpty()
Size()
Keys()



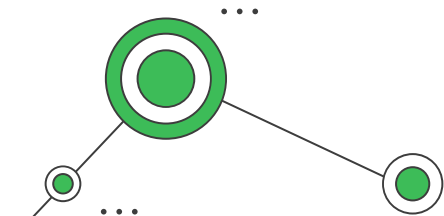
Hashing procedures will give us big integers

Use modular arithmetic to map their values to specific ranges.

...



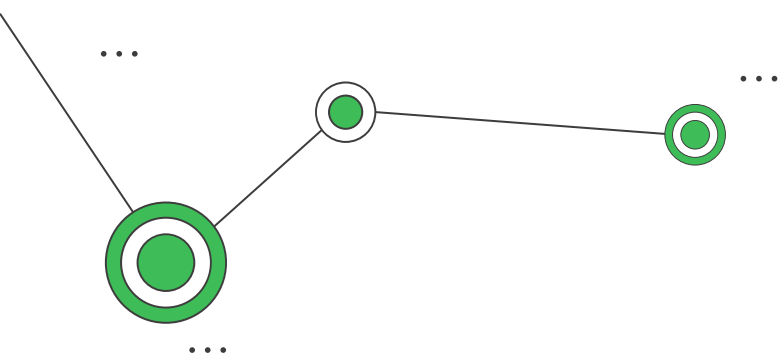
Hash Function

$$h(k: \mathbb{Z})$$


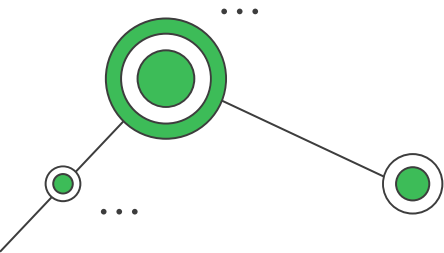
$$h: U \rightarrow \{0, 1, 2, \dots, m - 1\}$$

Requirements:

- $h(k)$ always returns the same value unless k changes.
- If $k_1 = k_2$, then $h(k_1) = h(k_2)$. Is the converse true?
- If $h(k_1) \neq h(k_2)$, then $k_1 \neq k_2$.
- Efficient to compute.
- Distribute keys uniformly.
- **"All the bits of the key play an equal role in computing every hash value."**



Division Method

$$h(k) = k \bmod m$$


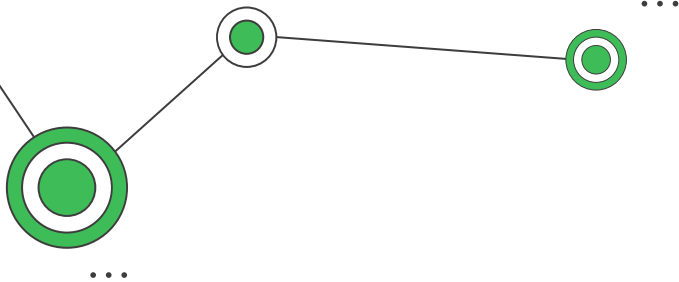
Goal: map a non-negative integer key k into one of the m slots in the table.

WARNING!!!

- If m is even, then $h(k)$ is odd or even if k is also odd or even, respectively.
- **Avoid $m = 2^p$. Otherwise, we are only using the p lowest-order bits of k .**
- **Avoid $m = 10^p$ if all keys are decimal numbers.**

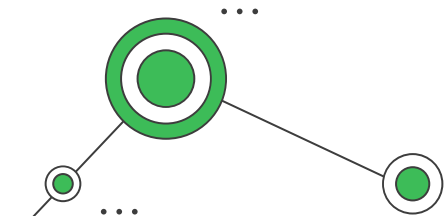
Better:

- Let m be a prime not too close to an exact power of 2.
- Check your $h(k)$ distributes uniformly in the table.



Multiplication Method

$$h(k) = \lfloor m(kA \bmod 1) \rfloor$$



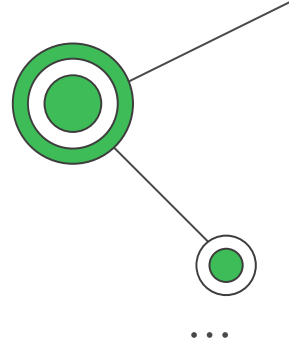
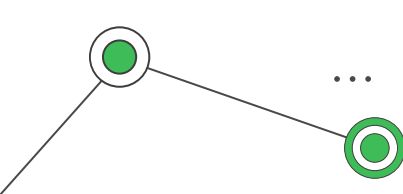
Goal: map a non-negative integer key k into one of the m slots in the table.

- Constant A in range $0 < A < 1$.
- As expected, there are better values for A than others. Knuth suggests:

$$A \approx \frac{\sqrt{5} - 1}{2} = 0.6180339887$$

- The value of m is not that critical. We can use $m = 2^p$.
- **Still, check your $h(k)$ distributes uniformly in the table.**

Example



0	1	2	3	4	5	6	7	8	9	10
∅	∅	∅	∅	∅	∅	∅	∅	∅	∅	∅

Input:



$h(k)$: For a color name k , use the numerical position in the alphabet of the first letter in k .


$$h(\text{"yellow"}) = 25 \bmod 11 = 3$$





$$h(\text{"green"}) = 7 \bmod 11 = 7$$

$$h(\text{"purple"}) = 16 \bmod 11 = 5$$

$$h(\text{"blue"}) = 2 \bmod 11 = 2$$

Example



0	1	2	3	4	5	6	7	8	9	10
∅	∅			∅		∅		∅	∅	∅

Search("blue"):

$$h(\text{"blue"}) = 2 \bmod 11 = 2$$

Return $T[2]$

Delete():

$$h(\text{"green"}) = 7 \bmod 11 = 7$$

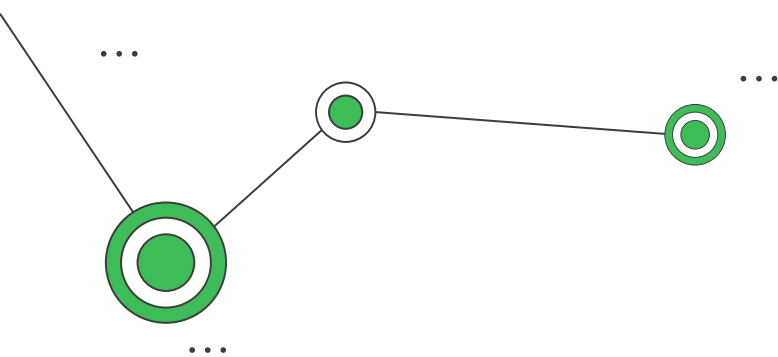
$T[7] = \text{null}$

Insert():

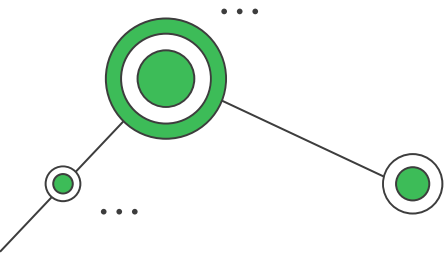
$$h(\text{"magenta"}) = 13 \bmod 11 = 2$$

But $T[2]$ has





Collisions



Let k_1 and k_2 be two different keys.
There is a collision in the hash table if $h(k_1) = h(k_2)$.

Ideal: Design a collision-free hash function.

Reality: Hard to design a $h(k)$ that creates random slot indices from non-random keys. **Assume collisions will occur.**

Solution: Implement **collision management strategies.**

Collision Management Strategies

01

Chaining

Multiple items in a slot?
Store them in a Doubly
Linked List.

02

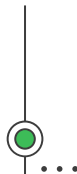
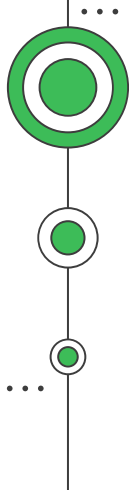
Open Addressing

Is the slot occupied?
Search for the next one
available.

02

Chaining

Store items in Linked Lists



Luhn, H.P. (1953), A new method of recording and searching information. Amer. Doc., 4: 14-16.

A NEW METHOD OF RECORDING AND SEARCHING INFORMATION

H. P. LUHN*

This method applies to the procedures required to record a legend concerning a document and to enable an inquirer to locate this document by means of the legend, if it is related to a specified subject.

The conventional methods of indexing and classifying attempt to evaluate the relative importance of a plurality of aspects contained in a document and makes the most important one the key for locating the document within an orderly scale of a certain dimension. Subordinated aspects are covered by way of reference in appropriate other locations of the scale.

One of the disadvantages of the conventional system is that the standard of value on which the indexer bases his decision may change and, what suddenly is considered an aspect of major significance, may not have been included in the classification or index at the time, even though it was contained in a document.

Another drawback is that it becomes difficult for an inquirer to reverse the process of classification or indexing and pose his query in a form matching to a reasonable degree the values of a potential reference.

The new method uses the principle of characterizing a topic by a set of identifying elements or criteria. These elements may be of any dimension and as many may be recorded as is desirable. Also, they are not weighted and no significance need be implied by the order in which they are given.

One of the main functions of the new method is that of producing a response to an inquiry in all cases, even if the reference appears to be remote, it being the understanding that it is the closest available.

The elements enumerated by recorders to identify a topic will necessarily vary as no two recorders will view a topic in identical fashion. Similarly, no two inquirers, when referring to the same subject will state their query in identical

fashion. It is therefore important that a system recognizes that these variations arise and that they cannot be controlled. It must then become the function of the system to overcome these variations to a reasonable degree.

When identifying a topic by a set of criteria or identifying terms, the more terms are stated the more specifically the topic is delineated. Each term in turn may be a concept which is itself may vary as to specificity. If we consider a concept as being a field in a multi-dimensional array, we may then visualize a topic as being located in that space which is common to all the concept fields stated. It may further be visualized that related topics are located more or less adjacent to each other depending on the degree of similarity and that this is so because they agree in some of the identifying terms and therefore share some of the concept fields. Figure 1 is a diagrammatic illustration.

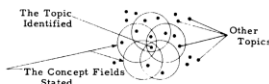


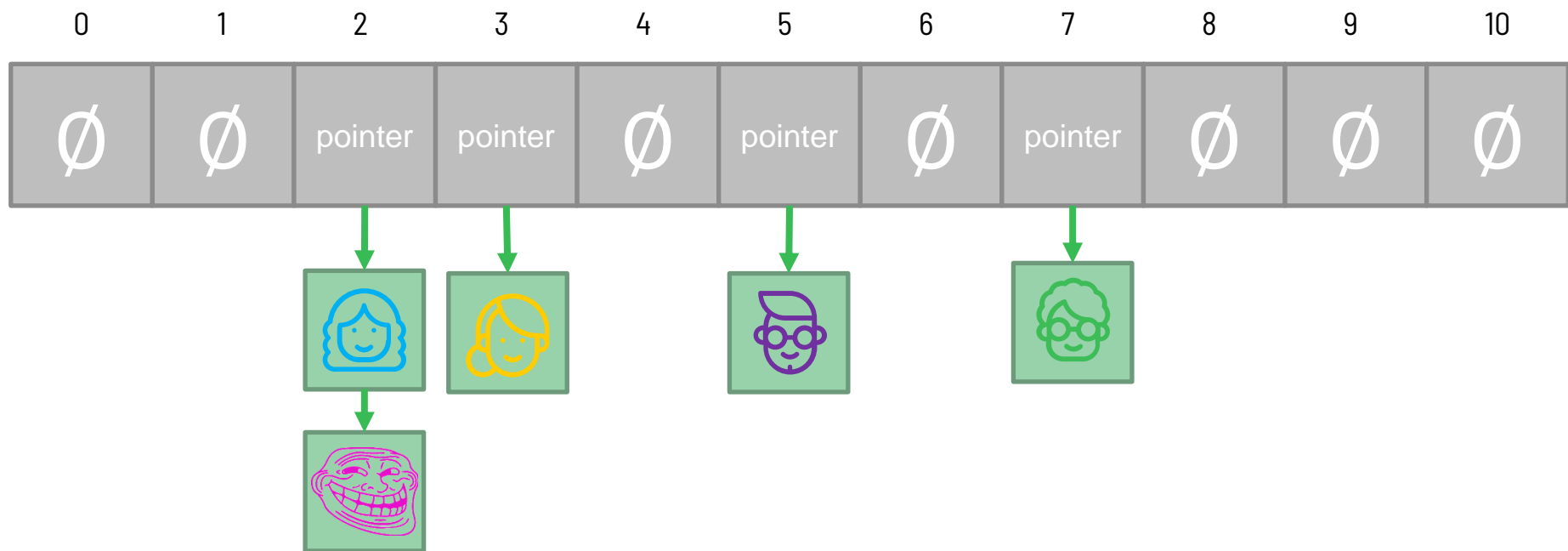
FIGURE 1.

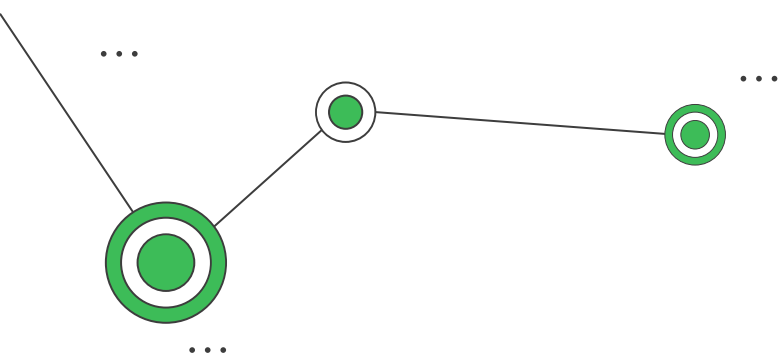
In order to understand the nature of the arrangement, let us assume a vocabulary of 100 concepts and let us identify a topic by five conceptual terms. By using all possible combinations of five terms, a total of 75 million patterns of criteria result, each of these patterns having a fixed location within the system. If then a topic is identified by five terms of the vocabulary, it is thereby assigned to a definite one of these fixed locations.

While assuming that there is an ideal and true location where a topic belongs, it is un-

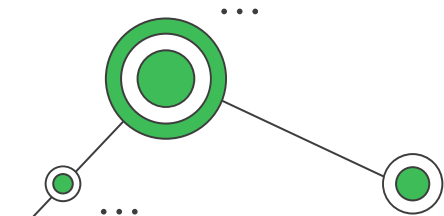
*International Business Machines Corporation, Engineering Laboratory, Poughkeepsie, New York.

Example: Chaining





Chaining: Analysis



Let n be the number of keys, and m the number of slots.

- **Worst-case: Traversing all items in the table is $O(n)$**
- **Simple Uniform Hashing Assumption:** each key is equally likely to be hashed to any slot in the table independent of other keys hashing.
- If $h(k)$ distributes keys randomly, then the expected length of a list is $\alpha = n/m$ (aka. **Load Factor**).
- Runtime: $\Theta(1 + |\text{chain}|) = \Theta(1 + \alpha)$
- If $n \in O(m)$, $\alpha = \frac{n}{m} = \frac{O(m)}{m} = O(1)$

Do not get confused!



Hashing: a process or technique used to convert input data of any size into a fixed-size value or key.

Hash function: a specific algorithm used in hashing to map data of arbitrary size to data of a fixed size.

Hash table: a data structure that implements an associative array abstract data type, a structure that can map keys to values.

$h(\text{last slide}) = \text{End}$

Do you have any questions?

CREDITS: This presentation template was created by [Slidesgo](#), including icons by [Flaticon](#), infographics & images by [Freepik](#) and illustrations by [Stories](#)