

Developing an Algorithm for the "Wildcard" Problem

When working out the steps of an algorithm, it is helpful to investigate several examples.

Case 1: No "*" in Wildcard String

0	1	2	3	4	5	6	7
d	e	?	o	?	?	c	e
d	e	n	o	u	n	c	e

If the "wildcard" string does not contain any "*" characters (ie. contains ONLY "?" characters)

- If the string lengths are NOT equal, the strings DO NOT match
- If the string lengths are equal, scan the strings character by character, stopping as soon as corresponding characters fail to match (recall that "?" matches any character) or when the end of the string is reached
- If the end of the string is reached and no characters fail to match, then the two strings match. Otherwise, they do not match.

Case 2: The Wildcard String Contains "*"

0	1	2	3	4	5	6	7
d ✓	*	u ✓	*	e ✓			
d ✓	e	n	o	u ✓	n	c	e ✓

⏟
* ⏟
 *

This case is more complex than case 1.
A great deal of thought is required to
understand this case thoroughly.

∴ you must be PATIENT
and PERSEVERANT

MORAL OF THE STORY

- Break down large, complex problems into smaller, simpler subproblems
- Think of Michelangelo chipping away at a slab of rock.

He thought of the sculpture already existing within the slab. His job was to chip away and "find" it.

Problem solving is much the same.
A great deal of "chipping away" is
required before a solution emerges.