String Searching

Saikrishna Arcot
(edits by M. Hudachek-Buswell)

March 27, 2017

String Searching

- A string is a sequence of characters, and an alphabet is the set of possible characters in a family of strings

String Searching

- A string is a sequence of characters, and an alphabet is the set of possible characters in a family of strings
- The concept of string searching, or pattern matching, is where you look to see if a substring, *pattern*, is within a larger string, *text*.

String Searching

- A string is a sequence of characters, and an alphabet is the set of possible characters in a family of strings
- The concept of string searching, or pattern matching, is where you look to see if a substring, *pattern*, is within a larger string, *text*.
- We refer to the length of the *text* as n, and the length of the *pattern* as m

String Searching

- A string is a sequence of characters, and an alphabet is the set of possible characters in a family of strings
- The concept of string searching, or pattern matching, is where you look to see if a substring, *pattern*, is within a larger string, *text*.
- We refer to the length of the *text* as n, and the length of the *pattern* as m
- There are three algorithms we explore: Brute Force, Boyer-Moore, and KMP

String Searching

- A string is a sequence of characters, and an alphabet is the set of possible characters in a family of strings
- The concept of string searching, or pattern matching, is where you look to see if a substring, *pattern*, is within a larger string, *text*.
- We refer to the length of the *text* as n, and the length of the *pattern* as m
- There are three algorithms we explore: Brute Force, Boyer-Moore, and KMP
- In terms of efficiency in pattern matching algorithms, we look at the number of comparisons.

String Searching

- A string is a sequence of characters, and an alphabet is the set of possible characters in a family of strings
- The concept of string searching, or pattern matching, is where you look to see if a substring, *pattern*, is within a larger string, *text*.
- We refer to the length of the *text* as n, and the length of the *pattern* as m
- There are three algorithms we explore: Brute Force, Boyer-Moore, and KMP
- In terms of efficiency in pattern matching algorithms, we look at the number of comparisons.
- There are many applications for these algorithms: test editors, search engines, and DNA and image sequences.

Brute Force

- Brute force is the simplest and least efficient string searching algorithm.

Brute Force

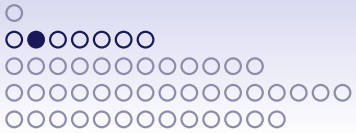
- Brute force is the simplest and least efficient string searching algorithm.
- Align the pattern at the beginning of the text.

Brute Force

- Brute force is the simplest and least efficient string searching algorithm.
- Align the pattern at the beginning of the text.
- Compare the first character in the text with the first character in the pattern.

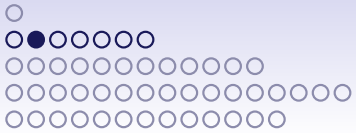
Brute Force

- Brute force is the simplest and least efficient string searching algorithm.
- Align the pattern at the beginning of the text.
- Compare the first character in the text with the first character in the pattern.
- If they don't match, then shift the pattern to the right by one character and compare the first character in the pattern with the current character in the text. If they do match, compare the next character in the pattern with the next character in the text. Repeat this step until you find a match or there can be no more matches.



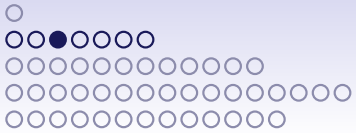
Brute Force

- Once a match is found, to search for additional matches, shift the pattern to the right by one, and compare the first character in the pattern to the current character in the text.



Brute Force

- Once a match is found, to search for additional matches, shift the pattern to the right by one, and compare the first character in the pattern to the current character in the text.
- If any part of the pattern is "hanging off" of the end of the text, then there are no more matches.



Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						

Brute Force

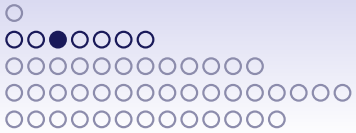
For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						



Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				
			a	c	c			

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				
			a	c	c			

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				
			a	c	c			

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				
			a	c	c			

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				
			a	c	c			
				a	c	c		

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				
			a	c	c			
				a	c	c		

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				
			a	c	c			
				a	c	c		
					a	c	c	

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				
			a	c	c			
				a	c	c		
					a	c	c	

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				
			a	c	c			
				a	c	c		
					a	c	c	

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				
			a	c	c			
				a	c	c		
					a	c	c	

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				
			a	c	c			
				a	c	c		
					a	c	c	

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				
			a	c	c			
				a	c	c		
					a	c	c	
						a	c	c

If you were looking for more matches, then the pattern would be shifted over by one character.

Brute Force

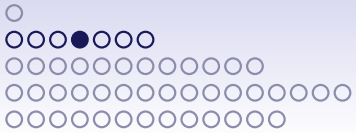
For example, if the text is “abbacacca” and the pattern is “acc”:

a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				
			a	c	c			
				a	c	c		
					a	c	c	
						a	c	c

Brute Force

For example, if the text is “abbacacca” and the pattern is “acc”:

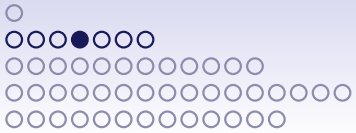
a	b	b	a	c	a	c	c	a
a	c	c						
	a	c	c					
		a	c	c				
			a	c	c			
				a	c	c		
					a	c	c	
						a	c	c



Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						



Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						

Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					

Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					

Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					
		b	u	i	l	d				

Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					
		b	u	i	l	d				

Brute Force

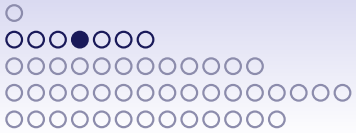
For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					
		b	u	i	l	d				
			b	u	i	l	d			

Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					
		b	u	i	l	d				
			b	u	i	l	d			



Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					
		b	u	i	l	d				
			b	u	i	l	d			

Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					
		b	u	i	l	d				
			b	u	i	l	d			
				b	u	i	l	d		

Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					
		b	u	i	l	d				
			b	u	i	l	d			
				b	u	i	l	d		

Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					
		b	u	i	l	d				
			b	u	i	l	d			
				b	u	i	l	d		

Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					
		b	u	i	l	d				
			b	u	i	l	d			
				b	u	i	l	d		

Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					
		b	u	i	l	d				
			b	u	i	l	d			
				b	u	i	l	d		
					b	u	i	l	d	

Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					
		b	u	i	l	d				
			b	u	i	l	d			
				b	u	i	l	d		
					b	u	i	l	d	

Brute Force

For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					
		b	u	i	l	d				
			b	u	i	l	d			
				b	u	i	l	d		
					b	u	i	l	d	
						b	u	i	l	d

Brute Force

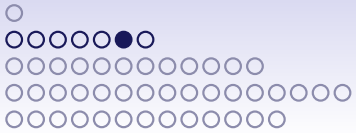
For example, if the text is “mythbusters” and the pattern is “build”:

m	y	t	h	b	u	s	t	e	r	s
b	u	i	l	d						
	b	u	i	l	d					
		b	u	i	l	d				
			b	u	i	l	d			
				b	u	i	l	d		
					b	u	i	l	d	
						b	u	i	l	d

Brute Force

(Note: the pseudocode below returns the first match.)

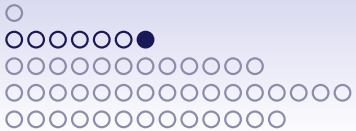
```
procedure BruteForce(text, pattern)
    i ← 0
    while i <= length of text − length of pattern do
        j ← 0
        while j < length of pattern and text[i + j] = pattern[j]
do
            j ← j + 1
        end while
```



Brute Force

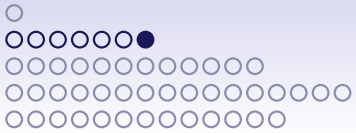
(Note: the pseudocode below returns the first match.)

```
        if j = length of pattern then
            return j
        else
            i ← i + 1
        end if
    end while
    return -1
end procedure
```



Performance

- In the best case, when searching for only the first match, brute force is $O(m)$, where m is the length of the pattern.



Performance

- In the best case, when searching for only the first match, brute force is $O(m)$, where m is the length of the pattern.
- In all other cases (best case when searching for all matches, as well as average and worst cases), brute force is $O(mn)$, where m is the length of the pattern and n is the length of the text, because there could be a mismatch on the very last character of the pattern at each alignment in the text.

Boyer-Moore

- Boyer-Moore string searching constructs a **last table** to determine how much to shift the pattern by on a mismatch.

Boyer-Moore

- Boyer-Moore string searching constructs a **last table** to determine how much to shift the pattern by on a mismatch.
- In addition, the algorithm starts from the back of the pattern instead of the front.

Boyer-Moore Last Table

- The last table used by Boyer-Moore is a mapping from each character in the alphabet (the set of all characters that may be in the pattern or the text) to the last index the character appears in the pattern.

Boyer-Moore Last Table

- The last table used by Boyer-Moore is a mapping from each character in the alphabet (the set of all characters that may be in the pattern or the text) to the last index the character appears in the pattern.
- If the character isn't in the pattern, then -1 is used as the value.

Boyer-Moore Last Table

- The last table used by Boyer-Moore is a mapping from each character in the alphabet (the set of all characters that may be in the pattern or the text) to the last index the character appears in the pattern.
- If the character isn't in the pattern, then -1 is used as the value.
- When writing out the last table, instead of writing potentially hundreds of entries for characters that aren't in the pattern, * can be used to represent all other characters.

Boyer-Moore Last Table

For example, the last table for “dog” is:

0	1	2
d	o	g

*
-1

Boyer-Moore Last Table

For example, the last table for “dog” is:

0	1	2
d	o	g

*
-1

Boyer-Moore Last Table

For example, the last table for “dog” is:

0	1	2
d	o	g

d	*
0	-1

Boyer-Moore Last Table

For example, the last table for “dog” is:

0	1	2
d	o	g

d	o	*
0	1	-1

Boyer-Moore Last Table

For example, the last table for “dog” is:

0	1	2
d	o	g

d	g	o	*
0	2	1	-1

Boyer-Moore Last Table

For example, the last table for “Dig-Dug” is:

0	1	2	3	4	5	6
D	i	g	-	D	u	g

*
-1

Boyer-Moore Last Table

For example, the last table for “Dig-Dug” is:

0	1	2	3	4	5	6
D	i	g	-	D	u	g

*
-1

Boyer-Moore Last Table

For example, the last table for “Dig-Dug” is:

0	1	2	3	4	5	6
D	i	g	-	D	u	g

D	*
0	-1

Boyer-Moore Last Table

For example, the last table for “Dig-Dug” is:

0	1	2	3	4	5	6
D	i	g	-	D	u	g

D	i	*
0	1	-1

Boyer-Moore Last Table

For example, the last table for “Dig-Dug” is:

0	1	2	3	4	5	6
D	i	g	-	D	u	g

D	g	i	*
0	2	1	-1

Boyer-Moore Last Table

For example, the last table for “Dig-Dug” is:

0	1	2	3	4	5	6
D	i	g	-	D	u	g

D	g	i	-	*
0	2	1	3	-1

Boyer-Moore Last Table

For example, the last table for “Dig-Dug” is:

0	1	2	3	4	5	6
D	i	g	-	D	u	g

D	g	i	-	*
4	2	1	3	-1

Boyer-Moore Last Table

For example, the last table for “Dig-Dug” is:

0	1	2	3	4	5	6
D	i	g	-	D	u	g

D	g	i	u	-	*
4	2	1	5	3	-1

Boyer-Moore Last Table

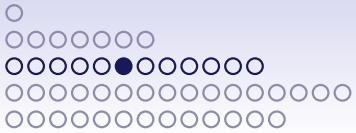
For example, the last table for “Dig-Dug” is:

0	1	2	3	4	5	6
D	i	g	-	D	u	g

D	g	i	u	-	*
4	6	1	5	3	-1

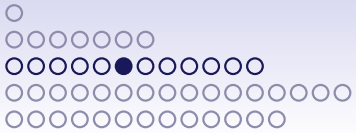
Boyer-Moore Last Table

procedure BoyerMooreLastTable(*pattern*)
    lastTable ← mapping from letter to integer
    for *i* ← 0, length of *pattern* **do**
        lastTable[*pattern*[*i*]] ← *i*
    end for
    return *lastTable*
end procedure



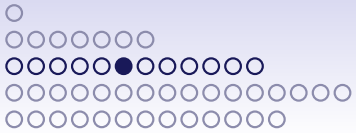
Boyer-Moore

- First, construct a last table for the pattern. The table should contain the last index each character in the alphabet is present in the pattern (or -1 if the character isn't in the pattern).



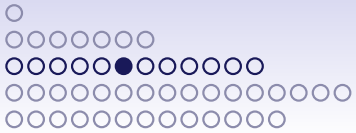
Boyer-Moore

- First, construct a last table for the pattern. The table should contain the last index each character in the alphabet is present in the pattern (or -1 if the character isn't in the pattern).
- Align the pattern at the beginning of the text.



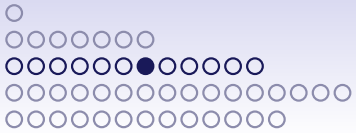
Boyer-Moore

- First, construct a last table for the pattern. The table should contain the last index each character in the alphabet is present in the pattern (or -1 if the character isn't in the pattern).
- Align the pattern at the beginning of the text.
- Compare the last character in the pattern with the character in the text at that index.



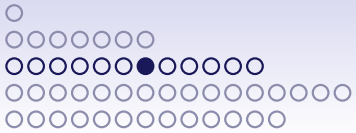
Boyer-Moore

- First, construct a last table for the pattern. The table should contain the last index each character in the alphabet is present in the pattern (or -1 if the character isn't in the pattern).
- Align the pattern at the beginning of the text.
- Compare the last character in the pattern with the character in the text at that index.
- If they do match, then go to the previous character in the pattern and the text, and repeat the previous step. If you've compared all of the characters in the pattern, you've found a match.



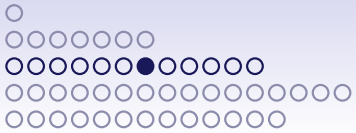
Boyer-Moore

- If they don't match, then take the character in the text, and look up the value in the last table to get the next alignment of the pattern. For example, if the value in the last table is 2, align the pattern so that index 2 of the pattern is at the mismatching character in the text. Compare again with the last character in the pattern and the current character in the text.



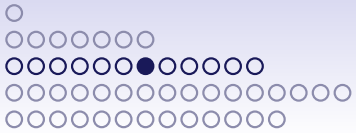
Boyer-Moore

- If they don't match, then take the character in the text, and look up the value in the last table to get the next alignment of the pattern. For example, if the value in the last table is 2, align the pattern so that index 2 of the pattern is at the mismatching character in the text. Compare again with the last character in the pattern and the current character in the text.
  - However, if this means the pattern would be shifted back (to the left), shift the pattern forward (to the right) instead by 1.



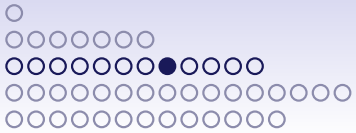
Boyer-Moore

- If they don't match, then take the character in the text, and look up the value in the last table to get the next alignment of the pattern. For example, if the value in the last table is 2, align the pattern so that index 2 of the pattern is at the mismatching character in the text. Compare again with the last character in the pattern and the current character in the text.
  - However, if this means the pattern would be shifted back (to the left), shift the pattern forward (to the right) instead by 1.
- Once a match is found, to search for additional matches, shift the pattern to the right by one, and compare the last character in the pattern to the current character in the text.



Boyer-Moore

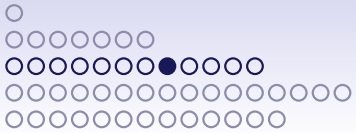
- If they don't match, then take the character in the text, and look up the value in the last table to get the next alignment of the pattern. For example, if the value in the last table is 2, align the pattern so that index 2 of the pattern is at the mismatching character in the text. Compare again with the last character in the pattern and the current character in the text.
  - However, if this means the pattern would be shifted back (to the left), shift the pattern forward (to the right) instead by 1.
- Once a match is found, to search for additional matches, shift the pattern to the right by one, and compare the last character in the pattern to the current character in the text.
- If any part of the pattern is "hanging off" of the end of the text, then there are no more matches.



Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						

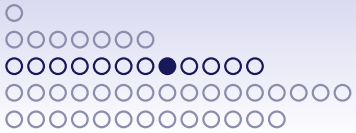


Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						

c	k	m	*
2	1	3	-1

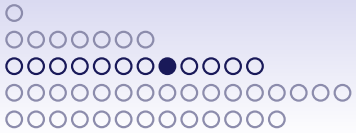


Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						

c	k	m	*
2	1	3	-1

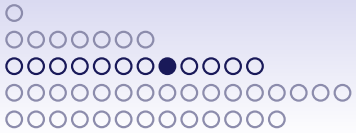


Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						

c	k	m	*
2	1	3	-1

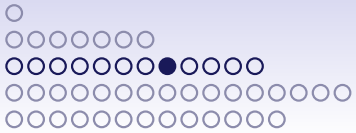


Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					

c	k	m	*
2	1	3	-1

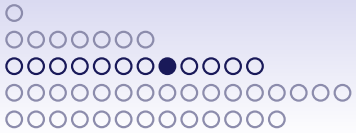


Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					

c	k	m	*
2	1	3	-1

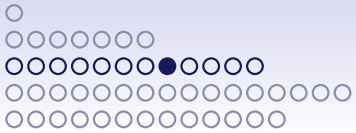


Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					

c	k	m	*
2	1	3	-1

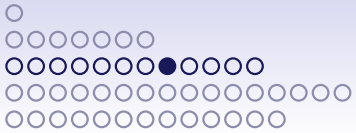


Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					
			c	k	c	m			

c	k	m	*
2	1	3	-1



Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					
			c	k	c	m			

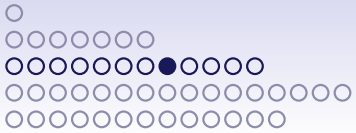
c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					
			c	k	c	m			

c	k	m	*
2	1	3	-1

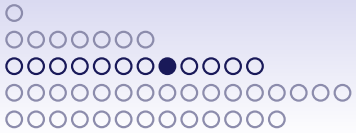


Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					
			c	k	c	m			
					c	k	c	m	

c	k	m	*
2	1	3	-1

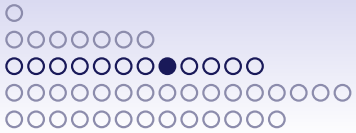


Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					
			c	k	c	m			
					c	k	c	m	

c	k	m	*
2	1	3	-1

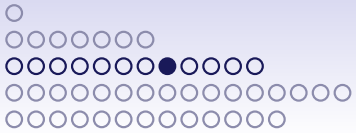


Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					
			c	k	c	m			
					c	k	c	m	

c	k	m	*
2	1	3	-1

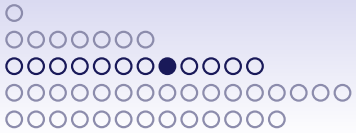


Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					
			c	k	c	m			
					c	k	c	m	

c	k	m	*
2	1	3	-1

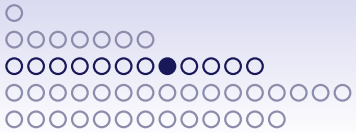


Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					
			c	k	c	m			
					c	k	c	m	

c	k	m	*
2	1	3	-1



Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					
			c	k	c	m			
					c	k	c	m	

c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					
			c	k	c	m			
					c	k	c	m	
						c	k	c	m

c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					
			c	k	c	m			
					c	k	c	m	
						c	k	c	m

c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					
			c	k	c	m			
					c	k	c	m	
						c	k	c	m

c	k	m	*
2	1	3	-1

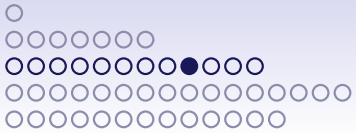
Boyer-Moore

For example, if the text is “abdckckcmd” and the pattern is “ckcm”:

a	b	d	c	k	c	k	c	m	d
c	k	c	m						
	c	k	c	m					
			c	k	c	m			
					c	k	c	m	
						c	k	c	m

c	k	m	*
2	1	3	-1

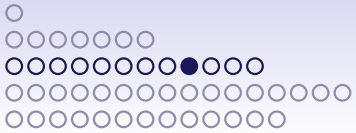
Next alignment is beyond the end of the text.



Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									

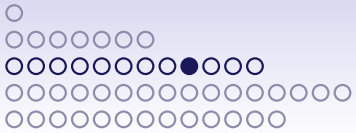


Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									

c	k	m	*
2	1	3	-1

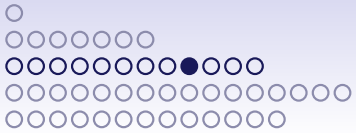


Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									

c	k	m	*
2	1	3	-1

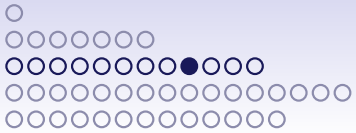


Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									

c	k	m	*
2	1	3	-1

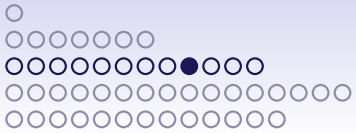


Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					

c	k	m	*
2	1	3	-1

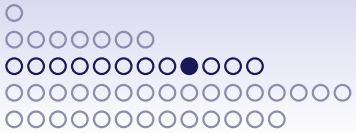


Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					

c	k	m	*
2	1	3	-1

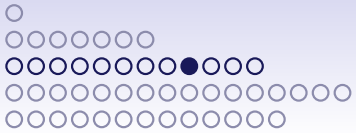


Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					

c	k	m	*
2	1	3	-1



Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					

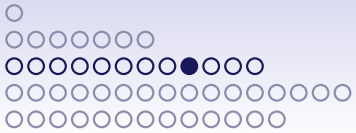
c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					

c	k	m	*
2	1	3	-1



Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					
			c	k	c	m						

c	k	m	*
2	1	3	-1

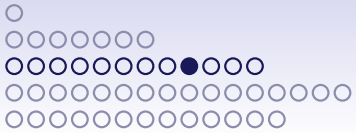
Never shift the pattern back.

Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					
					c	k	c	m				

c	k	m	*
2	1	3	-1



Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					
					c	k	c	m				

c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					
					c	k	c	m				

c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					
					c	k	c	m				
							c	k	c	m		

c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					
					c	k	c	m				
							c	k	c	m		

c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					
					c	k	c	m				
							c	k	c	m		

c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					
					c	k	c	m				
							c	k	c	m		
									c	k	c	m

c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					
					c	k	c	m				
							c	k	c	m		
									c	k	c	m

c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					
					c	k	c	m				
							c	k	c	m		
									c	k	c	m

c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					
					c	k	c	m				
							c	k	c	m		
									c	k	c	m

c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					
					c	k	c	m				
							c	k	c	m		
									c	k	c	m

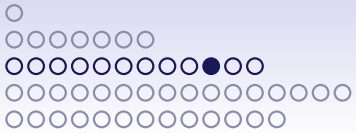
c	k	m	*
2	1	3	-1

Boyer-Moore

For example, if the text is “abdacccmkckcm” and the pattern is “ckcm”:

a	b	d	a	c	c	c	m	k	c	k	c	m
c	k	c	m									
				c	k	c	m					
					c	k	c	m				
							c	k	c	m		
									c	k	c	m

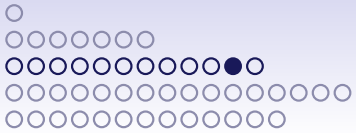
c	k	m	*
2	1	3	-1



Boyer-Moore

(Note: the pseudocode below returns the first match.)

```
procedure BoyerMoore(text, pattern)
    lastTable ← BoyerMooreLastTable(pattern)
    i ← 0
    while i <= length of text − length of pattern do
        j ← length of pattern − 1
        while j >= 0 and text[i + j] = pattern[j] do
            j ← j − 1
        end while
```



Boyer-Moore

(Note: the pseudocode below returns the first match.)

```
            if j = −1 then
                return i
            else
                shiftedIndex ← lastTable[text[i + j]]
                if shiftedIndex < j then
                    i ← i + (j − shiftedIndex)
                else
                    i ← i + 1
                end if
            end if
        end while
        return -1
    end procedure
```

Performance

- In the best case, when searching for only the first match, Boyer-Moore is $O(m)$, where m is the length of the pattern.

Performance

- In the best case, when searching for only the first match, Boyer-Moore is $O(m)$, where m is the length of the pattern.
- In the worst case, where the first character in the pattern is always a mismatching character, Boyer-Moore is $O(mn)$, where m is the length of the pattern and n is the length of the text.

Performance

- In the best case, when searching for only the first match, Boyer-Moore is $O(m)$, where m is the length of the pattern.
- In the worst case, where the first character in the pattern is always a mismatching character, Boyer-Moore is $O(mn)$, where m is the length of the pattern and n is the length of the text.
- In the average case, Boyer-Moore is $O(m+n)$, where m is the length of the pattern and n is the length of the text.

KMP

- Knuth-Morris-Pratt (KMP) string searching constructs a **failure table** (also known as a failure function) to determine how much to shift the pattern by on a mismatch.

KMP

- Knuth-Morris-Pratt (KMP) string searching constructs a **failure table** (also known as a failure function) to determine how much to shift the pattern by on a mismatch.
- KMP initially starts searching from the beginning of the pattern. However, when the pattern is shifted over, it may or may not restart from the beginning of the pattern

KMP Failure Table

- The failure table used by KMP is a table (or array) of the same length as the pattern. Each entry of the table contains a number representing the length of the longest suffix (up to that point) that is also a prefix in the pattern.

KMP Failure Table

- The failure table used by KMP is a table (or array) of the same length as the pattern. Each entry of the table contains a number representing the length of the longest suffix (up to that point) that is also a prefix in the pattern.
- In other words, given the word "revararev", the longest suffix here that is also a prefix is "rev" (notice how the word begins and ends with "rev", and that if we were to add another letter, the prefix "reva" wouldn't match the suffix "arev").

KMP Failure Table

- The failure table used by KMP is a table (or array) of the same length as the pattern. Each entry of the table contains a number representing the length of the longest suffix (up to that point) that is also a prefix in the pattern.
- In other words, given the word “revararev”, the longest suffix here that is also a prefix is “rev” (notice how the word begins and ends with “rev”, and that if we were to add another letter, the prefix “reva” wouldn’t match the suffix “arev”).
- For the word “ababaaababbabab”, the longest suffix here that is also a prefix is “abab”. Also note that there is also a shorter suffix that is also a prefix (“ab”), but the longest one should be used.

KMP Failure Table

- KMP needs to know the length of the longest suffix for the first 2 to m characters of the pattern (where m is the length of the pattern). In other words, for the word "revararev", this would be the following:

KMP Failure Table

- KMP needs to know the length of the longest suffix for the first 2 to m characters of the pattern (where m is the length of the pattern). In other words, for the word "revararev", this would be the following:
  - "re"

KMP Failure Table

- KMP needs to know the length of the longest suffix for the first 2 to m characters of the pattern (where m is the length of the pattern). In other words, for the word "revararev", this would be the following:
  - "re"
  - "rev"

KMP Failure Table

- KMP needs to know the length of the longest suffix for the first 2 to m characters of the pattern (where m is the length of the pattern). In other words, for the word "revararev", this would be the following:
  - "re"
  - "rev"
  - "reva"

KMP Failure Table

- KMP needs to know the length of the longest suffix for the first 2 to m characters of the pattern (where m is the length of the pattern). In other words, for the word "revararev", this would be the following:
  - "re"
  - "rev"
  - "reva"
  - "revar"

KMP Failure Table

- KMP needs to know the length of the longest suffix for the first 2 to m characters of the pattern (where m is the length of the pattern). In other words, for the word "revararev", this would be the following:
  - "re"
  - "rev"
  - "reva"
  - "revar"
  - "revara"

KMP Failure Table

- KMP needs to know the length of the longest suffix for the first 2 to m characters of the pattern (where m is the length of the pattern). In other words, for the word "revararev", this would be the following:
  - "re"
  - "rev"
  - "reva"
  - "revar"
  - "revara"
  - "revarar"

KMP Failure Table

- KMP needs to know the length of the longest suffix for the first 2 to m characters of the pattern (where m is the length of the pattern). In other words, for the word "revararev", this would be the following:
  - "re"
  - "rev"
  - "reva"
  - "revar"
  - "revara"
  - "revarar"
  - "revarare"

KMP Failure Table

- KMP needs to know the length of the longest suffix for the first 2 to m characters of the pattern (where m is the length of the pattern). In other words, for the word "revararev", this would be the following:
  - "re"
  - "rev"
  - "reva"
  - "revar"
  - "revara"
  - "revarar"
  - "revarare"
  - "revararev"

KMP Failure Table

- KMP needs to know the length of the longest suffix for the first 2 to m characters of the pattern (where m is the length of the pattern). In other words, for the word "revararev", this would be the following:
  - "re"
  - "rev"
  - "reva"
  - "revar"
  - "revara"
  - "revarar"
  - "revarare"
  - "revararev"
- Fortunately, there is a efficient way to do this in code.

KMP Failure Table

- Create two markers, `i` and `j`. `i` points to the first character in the pattern, while `j` points to the second character in the pattern. In the table, set the first entry to be 0.

KMP Failure Table

- Create two markers, `i` and `j`. `i` points to the first character in the pattern, while `j` points to the second character in the pattern. In the table, set the first entry to be 0.
- Compare the characters pointed to by `i` and `j`.

KMP Failure Table

- Create two markers, `i` and `j`. `i` points to the first character in the pattern, while `j` points to the second character in the pattern. In the table, set the first entry to be 0.
- Compare the characters pointed to by `i` and `j`.
  - If they are the same, write `i` + 1 into entry `j` of the table, and then move both `i` and `j` forward by one character.

KMP Failure Table

- Create two markers, `i` and `j`. `i` points to the first character in the pattern, while `j` points to the second character in the pattern. In the table, set the first entry to be 0.
- Compare the characters pointed to by `i` and `j`.
  - If they are the same, write `i` + 1 into entry `j` of the table, and then move both `i` and `j` forward by one character.
  - If they are different, and `i` is *not* at the first character of the pattern, then get the value at index `i - 1` of the table, and move `i` back to this value. Do not move `j`.

KMP Failure Table

- Create two markers, `i` and `j`. `i` points to the first character in the pattern, while `j` points to the second character in the pattern. In the table, set the first entry to be 0.
- Compare the characters pointed to by `i` and `j`.
  - If they are the same, write `i` + 1 into entry `j` of the table, and then move both `i` and `j` forward by one character.
  - If they are different, and `i` is *not* at the first character of the pattern, then get the value at index `i - 1` of the table, and move `i` back to this value. Do not move `j`.
  - If they are different, and `i` is at the first character of the pattern, then write 0 into entry `j` of the table, and move `j` forward by one character. Do not move `i`.

KMP Failure Table

- Create two markers, `i` and `j`. `i` points to the first character in the pattern, while `j` points to the second character in the pattern. In the table, set the first entry to be 0.
- Compare the characters pointed to by `i` and `j`.
  - If they are the same, write `i` + 1 into entry `j` of the table, and then move both `i` and `j` forward by one character.
  - If they are different, and `i` is *not* at the first character of the pattern, then get the value at index `i - 1` of the table, and move `i` back to this value. Do not move `j`.
  - If they are different, and `i` is at the first character of the pattern, then write 0 into entry `j` of the table, and move `j` forward by one character. Do not move `i`.
- Repeat the previous step until `j` goes past the end of the string, and all of the entries in the table have a value.

KMP Failure Table

For example, the failure table for “amanama” is:

0	1	2	3	4	5	6
a	m	a	n	a	m	a

KMP Failure Table

For example, the failure table for “amanama” is:

i	j					
0	1	2	3	4	5	6
a	m	a	n	a	m	a
0						

KMP Failure Table

For example, the failure table for “amanama” is:

i		j				
0	1	2	3	4	5	6
a	m	a	n	a	m	a
0	0					

KMP Failure Table

For example, the failure table for “amanama” is:

	i		j			
0	1	2	3	4	5	6
a	m	a	n	a	m	a
0	0	1				

KMP Failure Table

For example, the failure table for “amanama” is:

i			j			
0	1	2	3	4	5	6
a	m	a	n	a	m	a
0	0	1				

KMP Failure Table

For example, the failure table for “amanama” is:

i				j		
0	1	2	3	4	5	6
a	m	a	n	a	m	a
0	0	1	0			

KMP Failure Table

For example, the failure table for “amanama” is:

	i				j	
0	1	2	3	4	5	6
a	m	a	n	a	m	a
0	0	1	0	1		

KMP Failure Table

For example, the failure table for “amanama” is:

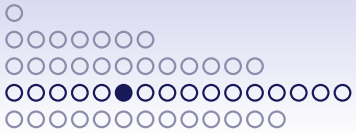
		i				j
0	1	2	3	4	5	6
a	m	a	n	a	m	a
0	0	1	0	1	2	

KMP Failure Table

For example, the failure table for “amanama” is:

			i			
0	1	2	3	4	5	6
a	m	a	n	a	m	a
0	0	1	0	1	2	3

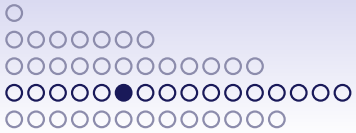
j would be pointing to index 7, which doesn’t exist.



KMP Failure Table

For example, the failure table for “andanandandan” is:

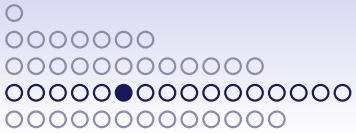
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n



KMP Failure Table

For example, the failure table for “andanandandan” is:

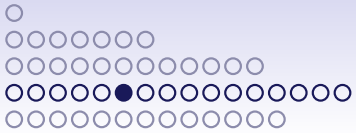
i	j											
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0												



KMP Failure Table

For example, the failure table for “andanandandan” is:

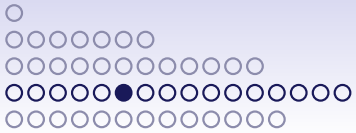
i		j										
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0	0											



KMP Failure Table

For example, the failure table for “andanandandan” is:

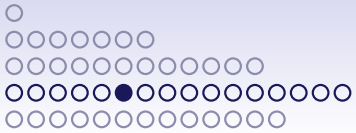
i			j									
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0	0	0										



KMP Failure Table

For example, the failure table for “andanandandan” is:

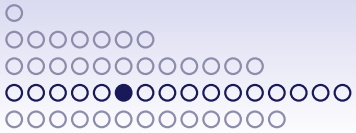
	i			j								
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0	0	0	1									



KMP Failure Table

For example, the failure table for “andanandandan” is:

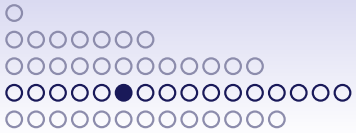
		i			j							
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0	0	0	1	2								



KMP Failure Table

For example, the failure table for “andanandandan” is:

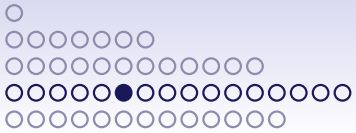
i					j							
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0	0	0	1	2								



KMP Failure Table

For example, the failure table for “andanandandan” is:

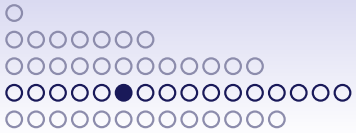
	i					j						
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0	0	0	1	2	1							



KMP Failure Table

For example, the failure table for “andanandandan” is:

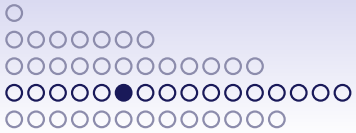
		i					j					
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0	0	0	1	2	1	2						



KMP Failure Table

For example, the failure table for “andanandandan” is:

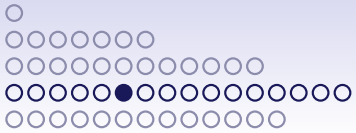
			i					j				
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0	0	0	1	2	1	2	3					



KMP Failure Table

For example, the failure table for “andanandandan” is:

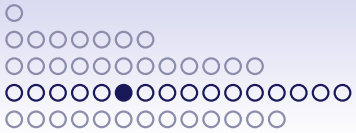
				i					j			
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0	0	0	1	2	1	2	3	4				



KMP Failure Table

For example, the failure table for “andanandandan” is:

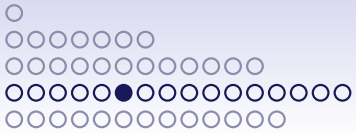
					i					j		
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0	0	0	1	2	1	2	3	4	5			



KMP Failure Table

For example, the failure table for “andanandandan” is:

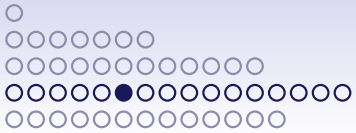
		i								j		
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0	0	0	1	2	1	2	3	4	5			



KMP Failure Table

For example, the failure table for “andanandandan” is:

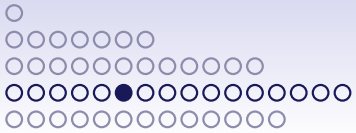
			i								j	
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0	0	0	1	2	1	2	3	4	5	3		



KMP Failure Table

For example, the failure table for “andanandandan” is:

				i								j
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0	0	0	1	2	1	2	3	4	5	3	4	

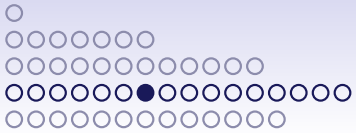


KMP Failure Table

For example, the failure table for “andanandandan” is:

					i							
0	1	2	3	4	5	6	7	8	9	10	11	12
a	n	d	a	n	a	n	d	a	n	d	a	n
0	0	0	1	2	1	2	3	4	5	3	4	5

j points to index 13.

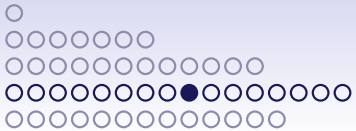


KMP Failure Table

```
procedure KMPFailureTable(pattern)
    failureTable ← array that is the same length as pattern
    i ← 0
    j ← 1
    failureTable[0] ← 0
    while j < length of pattern do
        if pattern[i] = pattern[j] then
            i ← i + 1
            failureTable[j] ← i
            j ← j + 1
        else
```

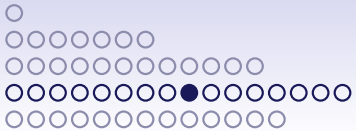
KMP Failure Table

```
            if i = 0 then
                failureTable[j] ← 0
                j ← j + 1
            else
                i ← failureTable[i − 1]
            end if
        end if
    end while
    return failureTable
end procedure
```



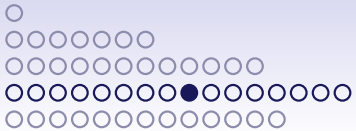
KMP

- First, construct a failure table for the pattern. The table should contain the length of the longest prefix that is the same as the suffix (of the same length) up to each character.



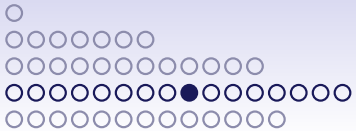
KMP

- First, construct a failure table for the pattern. The table should contain the length of the longest prefix that is the same as the suffix (of the same length) up to each character.
- Align the pattern at the beginning of the text.



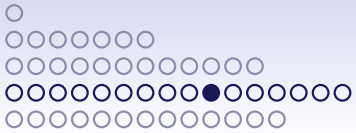
KMP

- First, construct a failure table for the pattern. The table should contain the length of the longest prefix that is the same as the suffix (of the same length) up to each character.
- Align the pattern at the beginning of the text.
- Compare the first character in the pattern with the character in the text at that index.



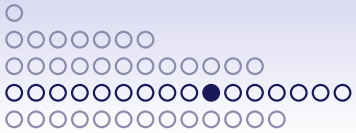
KMP

- First, construct a failure table for the pattern. The table should contain the length of the longest prefix that is the same as the suffix (of the same length) up to each character.
- Align the pattern at the beginning of the text.
- Compare the first character in the pattern with the character in the text at that index.
- If they do match, then go to the next character in the pattern and the text, and repeat the previous step. If you've compared all of the characters in the pattern, you've found a match.



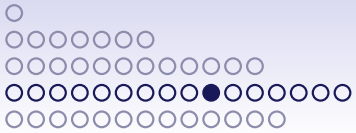
KMP

- If they don't match:



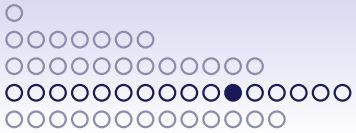
KMP

- If they don't match:
  - and the mismatch is on the first letter of the pattern, then shift the pattern to the right by one and restart comparing the pattern and the text from the first letter of the pattern.



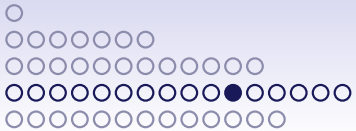
KMP

- If they don't match:
  - and the mismatch is on the first letter of the pattern, then shift the pattern to the right by one and restart comparing the pattern and the text from the first letter of the pattern.
  - and the mismatch is *not* on the first letter of the pattern, then, use the failure table to determine how much to shift the pattern by. Assuming the mismatch was on index j of the pattern, look at index $j-1$ of the failure table. This value tells you the next alignment of the pattern with the text. Align the pattern such that index $failureTable[j-1]$ of the pattern is aligned with the mismatching character in the text. Then, continue comparing from index $failureTable[j-1]$ of the pattern. Do not restart from the beginning.



KMP

- Once a match is found, to search for additional matches, look at the last index of the failure table. This value tells you the next alignment of the pattern with the text. Align the pattern such that the letter represented by this value is aligned **after** the last letter in the text. Continue comparing from this place in the pattern.



KMP

- Once a match is found, to search for additional matches, look at the last index of the failure table. This value tells you the next alignment of the pattern with the text. Align the pattern such that the letter represented by this value is aligned **after** the last letter in the text. Continue comparing from this place in the pattern.
- If any part of the pattern is "hanging off" of the end of the text, then there are no more matches.

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					
			e	a	s	e	e				

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					
			e	a	s	e	e				

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					
			e	a	s	e	e				

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					
			e	a	s	e	e				

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					
			e	a	s	e	e				

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					
			e	a	s	e	e				

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					
			e	a	s	e	e				

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					
			e	a	s	e	e				

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					
			e	a	s	e	e				
							e	a	s	e	e

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					
			e	a	s	e	e				
							e	a	s	e	e

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					
			e	a	s	e	e				
							e	a	s	e	e

e	a	s	e	e
0	0	0	1	1

KMP

For example, if the text is “ealeaseealle” and the pattern is “easee”:

e	a	l	e	a	s	e	e	a	l	l	e
e	a	s	e	e							
		e	a	s	e	e					
			e	a	s	e	e				
							e	a	s	e	e

e	a	s	e	e
0	0	0	1	1

Next alignment causes the pattern to hang off of the end of the text.

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						
										a	b	a	b	c	c	a	b	a	b

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						
										a	b	a	b	c	c	a	b	a	b

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						
										a	b	a	b	c	c	a	b	a	b

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						
										a	b	a	b	c	c	a	b	a	b

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						
										a	b	a	b	c	c	a	b	a	b

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						
										a	b	a	b	c	c	a	b	a	b

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						
										a	b	a	b	c	c	a	b	a	b

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

For example, if the text is “abababababccababccabab” and the pattern is “ababccabab”:

a	b	a	b	a	b	a	b	c	c	a	b	a	b	c	c	a	b	a	b
a	b	a	b	c	c	a	b	a	b										
		a	b	a	b	c	c	a	b	a	b								
				a	b	a	b	c	c	a	b	a	b						
										a	b	a	b	c	c	a	b	a	b

a	b	a	b	c	c	a	b	a	b
0	0	1	2	0	0	1	2	3	4

KMP

(Note: the pseudocode below returns **all** matches.)

```
procedure KMP(text, pattern)
    failureTable ← KMPFailureTable(pattern)
    matches ← array holding indices of matches
    i ← 0
    j ← 0
    while i <= length of text − length of pattern do
        while j < length of pattern and text[i + j] = pattern[j]
do
            j ← j + 1
        end while
```

KMP

(Note: the pseudocode below returns **all** matches.)

```
        if j = 0 then
            i ← i + 1
        else
            if j = length of pattern then
                Add i to matches
            end if
            nextAlignment ← failureTable[j − 1]
            i ← i + j − nextAlignment
            j ← nextAlignment
        end if
    end while
    return matches
end procedure
```

Performance

- In the best case, when searching for only the first match, KMP is $O(m)$, where m is the length of the pattern.

Performance

- In the best case, when searching for only the first match, KMP is $O(m)$, where m is the length of the pattern.
- In all other cases, KMP is $O(m+n)$, where m is the length of the pattern and n is the length of the text.

Rabin-Karp

- Rabin-Karp string searching doesn't directly compare characters (at least, initially).

Rabin-Karp

- Rabin-Karp string searching doesn't directly compare characters (at least, initially).
- It instead compares the hashes of the pattern and a substring of the text.

Rabin-Karp Rolling Hash

- Rabin-Karp uses a **rolling hash** to calculate a hash of the pattern and a hash of the substring (that is the same length as the pattern) of the text.

Rabin-Karp Rolling Hash

- Rabin-Karp uses a **rolling hash** to calculate a hash of the pattern and a hash of the substring (that is the same length as the pattern) of the text.
- If these hashes are not equal, then it is guaranteed that there is not a match starting at this index, and the rolling hash "slides" to the right by one character.

Rabin-Karp Rolling Hash

- Rabin-Karp uses a **rolling hash** to calculate a hash of the pattern and a hash of the substring (that is the same length as the pattern) of the text.
- If these hashes are not equal, then it is guaranteed that there is not a match starting at this index, and the rolling hash "slides" to the right by one character.
- If these hashes are equal, then there **may** be (but not guaranteed to be, because hashes can collide) a match starting at this index. At this point, each character is compared.

Rabin-Karp Rolling Hash

- Rabin-Karp uses a **rolling hash** to calculate a hash of the pattern and a hash of the substring (that is the same length as the pattern) of the text.
- If these hashes are not equal, then it is guaranteed that there is not a match starting at this index, and the rolling hash "slides" to the right by one character.
- If these hashes are equal, then there **may** be (but not guaranteed to be, because hashes can collide) a match starting at this index. At this point, each character is compared.
- A special property of a rolling hash is that while it is $O(n)$ to calculate the initial hash, it is $O(1)$ to "slide" the hash window and calculate the new hash.

Rabin-Karp Rolling Hash

- A rolling hash function that is commonly used is known as the **Rabin fingerprint**.

Rabin-Karp Rolling Hash

- A rolling hash function that is commonly used is known as the **Rabin fingerprint**.
- To calculate the hash, the following formula is used: $\sum_{i=0}^{j} text[i] \times BASE^{j-i}$, where j is the number of letters in $text$, and $BASE$ is a prime number.

Rabin-Karp Rolling Hash

- A rolling hash function that is commonly used is known as the **Rabin fingerprint**.
- To calculate the hash, the following formula is used: $\sum_{i=0}^{j} text[i] \times BASE^{j-i}$, where j is the number of letters in $text$, and $BASE$ is a prime number.
- Note that, in computing, each letter can be represented as an integer. In Java, the integer representation of each character is its Unicode value (It's actually a little more complicated than this, but this is sufficient for this algorithm.).

Rabin-Karp Rolling Hash

- Formula: $\sum_{i=0}^{j} text[i] \times BASE^{j-i-1}$, where j is the number of letters in $text$, and $BASE$ is a prime number.

Rabin-Karp Rolling Hash

- Formula: $\sum_{i=0}^{j} text[i] \times BASE^{j-i-1}$, where j is the number of letters in *text*, and *BASE* is a prime number.
- For the text (or the section of the text) you need to calculate the hash of, calculate the result of the formula. j is the number of letters that you need to hash, and *BASE* is a prime number that is usually provided.

Rabin-Karp Rolling Hash

For example, to calculate the hash of “crow”, given a base of 101: $\sum_{i=0}^{j} text[i] \times BASE^{j-i-1}$

Rabin-Karp Rolling Hash

For example, to calculate the hash of “crow”, given a base of 101:
$\sum_{i=0}^{j} text[i] \times BASE^{j-i-1}$
$'c' \times 101^3 +' r' \times 101^2 +' o' \times 101^1 +' w' \times 101^0$

Rabin-Karp Rolling Hash

For example, to calculate the hash of "crow", given a base of 101:

$\sum_{i=0}^{j} text[i] \times BASE^{j-i-1}$

$'c' \times 101^3 +' r' \times 101^2 +' o' \times 101^1 +' w' \times 101^0$

$99 \times 101^3 + 114 \times 101^2 + 111 \times 101 + 119$

Rabin-Karp Rolling Hash

For example, to calculate the hash of "crow", given a base of 101:

$\sum_{i=0}^{j} text[i] \times BASE^{j-i-1}$

$'c' \times 101^3 +' r' \times 101^2 +' o' \times 101^1 +' w' \times 101^0$

$99 \times 101^3 + 114 \times 101^2 + 111 \times 101 + 119$

$103,174,043$

Rabin-Karp Rolling Hash

For example, to calculate the hash of the first four characters of "welcome", given a base of 157:
$\sum_{i=0}^{j} text[i] \times BASE^{j-i-1}$

Rabin-Karp Rolling Hash

For example, to calculate the hash of the first four characters of "welcome", given a base of 157:

$\sum_{i=0}^{j} text[i] \times BASE^{j-i-1}$

$'w' \times 157^3 +' e' \times 157^2 +' l' \times 157^1 +' c' \times 157^0$

Rabin-Karp Rolling Hash

For example, to calculate the hash of the first four characters of "welcome", given a base of 157:

$\sum_{i=0}^{j} text[i] \times BASE^{j-i-1}$

$'w' \times 157^3 +' e' \times 157^2 +' l' \times 157^1 +' c' \times 157^0$

$119 \times 157^3 + 101 \times 157^2 + 108 \times 157 + 99$

Rabin-Karp Rolling Hash

For example, to calculate the hash of the first four characters of "welcome", given a base of 157:

$\sum_{i=0}^{j} text[i] \times BASE^{j-i-1}$

$'w' \times 157^3 +' e' \times 157^2 +' l' \times 157^1 +' c' \times 157^0$

$119 \times 157^3 + 101 \times 157^2 + 108 \times 157 + 99$

$463,023,871$

Rabin-Karp Rolling Hash

- Once you have the initial hash calculated, you can efficiently shift the window of text used for calculating the hash.

Rabin-Karp Rolling Hash

- Once you have the initial hash calculated, you can efficiently shift the window of text used for calculating the hash.
- For example, in the previous example, the hash for the first four characters (indices 0-3) of "welcome" was calculated. Using this hash, the hash for indices 1-4 can be calculated in $O(1)$ time.

Rabin-Karp Rolling Hash

- Once you have the initial hash calculated, you can efficiently shift the window of text used for calculating the hash.
- For example, in the previous example, the hash for the first four characters (indices 0-3) of "welcome" was calculated. Using this hash, the hash for indices 1-4 can be calculated in $O(1)$ time.
- Given a hash length of j, the character being removed from the hash at index a, and the character being added into the hash at index b, the new hash can be calculated using $BASE \times (oldHash - text[a] \times BASE^{j-1}) + text[b]$.

Rabin-Karp Rolling Hash

- Once you have the initial hash calculated, you can efficiently shift the window of text used for calculating the hash.
- For example, in the previous example, the hash for the first four characters (indices 0-3) of "welcome" was calculated. Using this hash, the hash for indices 1-4 can be calculated in $O(1)$ time.
- Given a hash length of j, the character being removed from the hash at index a, and the character being added into the hash at index b, the new hash can be calculated using $BASE \times (oldHash - text[a] \times BASE^{j-1}) + text[b]$.
- In other words, what is happening is that what the first letter contributed to the hash is being removed, all of the other letters are being shifted up by one slot, and the new letter is being added in.

Rabin-Karp Rolling Hash

For example, given the word “welcome”, the hash of 463,023,871 (this hash is for indices 0-3 of the word), and a base of 157, calculate the next hash:
$BASE \times (oldHash - text[a] \times BASE^{j-1}) + text[b]$

Rabin-Karp Rolling Hash

For example, given the word “welcome”, the hash of 463,023,871 (this hash is for indices 0-3 of the word), and a base of 157, calculate the next hash:

$BASE \times (oldHash - text[a] \times BASE^{j-1}) + text[b]$

$157 \times (463,023,871 -' w' \times 157^3) +' c'$

Rabin-Karp Rolling Hash

For example, given the word “welcome”, the hash of 463,023,871 (this hash is for indices 0-3 of the word), and a base of 157, calculate the next hash:

$BASE \times (oldHash - text[a] \times BASE^{j-1}) + text[b]$

$157 \times (463,023,871 -' w' \times 157^3) +' c'$

$157 \times (463,023,871 - 119 \times 157^3) + 99$

Rabin-Karp Rolling Hash

For example, given the word "welcome", the hash of 463,023,871 (this hash is for indices 0-3 of the word), and a base of 157, calculate the next hash:

$BASE \times (oldHash - text[a] \times BASE^{j-1}) + text[b]$

$157 \times (463,023,871 -' w' \times 157^3) +' c'$

$157 \times (463,023,871 - 119 \times 157^3) + 99$

$393,536,927$

Rabin-Karp

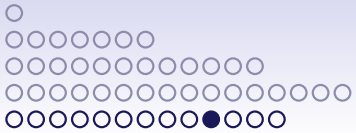
- For the string searching algorithm, first calculate the hash of the entire pattern and the hash of the first j characters of the text (where j is the length of the pattern).

Rabin-Karp

- For the string searching algorithm, first calculate the hash of the entire pattern and the hash of the first j characters of the text (where j is the length of the pattern).
- If the hashes are the same, compare each character in the pattern along with the characters included in the hash of the text. If all of the characters are the same, then you've found a match. Remember that two different strings may give the same hash.

Rabin-Karp

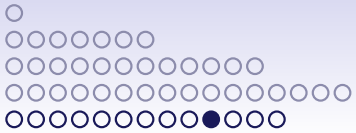
- For the string searching algorithm, first calculate the hash of the entire pattern and the hash of the first j characters of the text (where j is the length of the pattern).
- If the hashes are the same, compare each character in the pattern along with the characters included in the hash of the text. If all of the characters are the same, then you've found a match. Remember that two different strings may give the same hash.
- If the hashes are different, or if the characters don't match, slide the hash window of the text by one character to the right. (The hash of the pattern remains the same), and repeat the previous step until you've reached the end of the text.



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

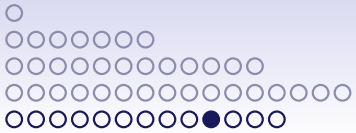
S	u	r	f	a	c	e		t	e	n	s	i	o	n
t	e	n	s											



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

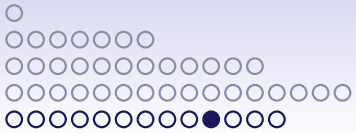
S	u	r	f	a	c	e		t	e	n	s	i	o	n
t	e	n	s											



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

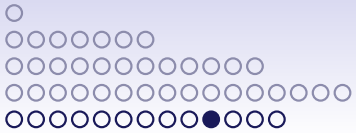
S	u	r	f	a	c	e		t	e	n	s	i	o	n
	t	e	n	s										



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

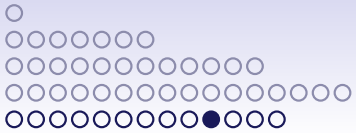
S	u	r	f	a	c	e		t	e	n	s	i	o	n
		t	e	n	s									



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

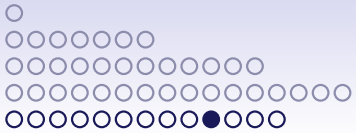
S	u	r	f	a	c	e		t	e	n	s	i	o	n
			t	e	n	s								



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

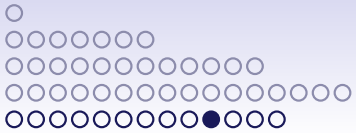
S	u	r	f	a	c	e		t	e	n	s	i	o	n
				t	e	n	s							



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

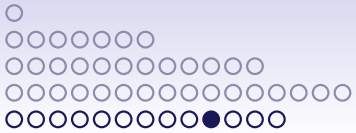
S	u	r	f	a	c	e		t	e	n	s	i	o	n
					t	e	n	s						



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

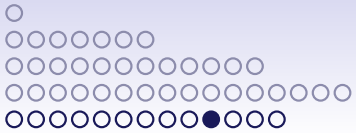
S	u	r	f	a	c	e		t	e	n	s	i	o	n
						t	e	n	s					



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

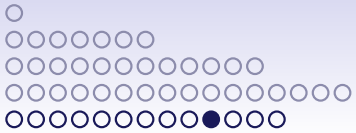
S	u	r	f	a	c	e		t	e	n	s	i	o	n
							t	e	n	s				



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

S	u	r	f	a	c	e		t	e	n	s	i	o	n
								t	e	n	s			

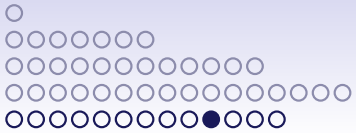


Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

S	u	r	f	a	c	e		t	e	n	s	i	o	n
								t	e	n	s			

Comparing characters

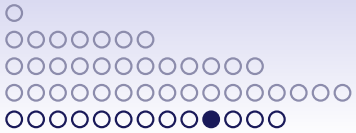


Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

S	u	r	f	a	c	e		t	e	n	s	i	o	n
								t	e	n	s			

Comparing characters

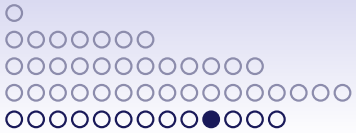


Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

S	u	r	f	a	c	e		t	e	n	s	i	o	n
								t	e	n	s			

Comparing characters

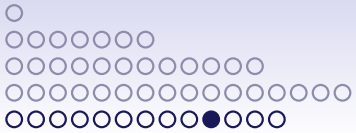


Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

S	u	r	f	a	c	e		t	e	n	s	i	o	n
								t	e	n	s			

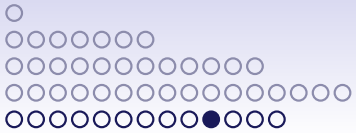
Comparing characters



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

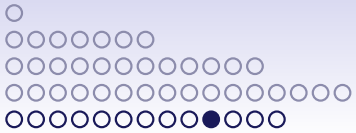
S	u	r	f	a	c	e		t	e	n	s	i	o	n
								t	e	n	s			



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

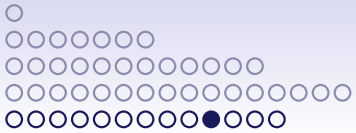
S	u	r	f	a	c	e		t	e	n	s	i	o	n
								t	e	n	s			
									t	e	n	s		



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

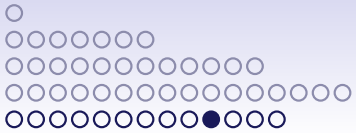
S	u	r	f	a	c	e		t	e	n	s	i	o	n
								t	e	n	s			
										t	e	n	s	



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

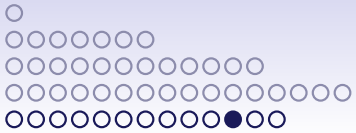
S	u	r	f	a	c	e		t	e	n	s	i	o	n
								t	e	n	s			
											t	e	n	s



Rabin-Karp

Because this algorithm is more mathematical than the others, not all of the detail is shown. The yellow cells represent the hash window currently being used.
For example, if the text is “Surface tension” and the pattern is “tens” (assume that, for this example, different strings give different hashes):

S	u	r	f	a	c	e		t	e	n	s	i	o	n
								t	e	n	s			
											t	e	n	s



Rabin-Karp

(Note: the pseudocode below returns only the first match.)

```
procedure RabinKarp(text, pattern)
    patternHash ← rolling hash of pattern
    textHash ← rolling hash of first pattern.length characters of
text
    i ← 0
    while i <= length of text − length of pattern do
        if patternHash = textHash then
            j ← 0
            while j < length of pattern and
text[i + j] = pattern[j] do
                j ← j + 1
            end while
```

Rabin-Karp

(Note: the pseudocode below returns only the first match.)

```
            if j = length of pattern then
                return i
            end if
        end if
        i ← i + 1
        if i <= length of text − length of pattern then
            textHash ← new hash of text, with the hash window
shifted over
        end if
    end while
    return -1
end procedure
```

Performance

- In the best case, when searching for only the first match, Rabin-Karp is $O(m)$, where m is the length of the pattern.

Performance

- In the best case, when searching for only the first match, Rabin-Karp is $O(m)$, where m is the length of the pattern.
- In the worst case, the pattern hash and the text hash are always equal (possibly due to a poor rolling hash function used), and each character is compared. In this case, Rabin-Karp is $O(mn)$, where m is the length of the pattern and n is the length of the text.

Performance

- In the best case, when searching for only the first match, Rabin-Karp is $O(m)$, where m is the length of the pattern.
- In the worst case, the pattern hash and the text hash are always equal (possibly due to a poor rolling hash function used), and each character is compared. In this case, Rabin-Karp is $O(mn)$, where m is the length of the pattern and n is the length of the text.
- In all other cases, Rabin-Karp is $O(m + n)$, where m is the length of the pattern and n is the length of the text.