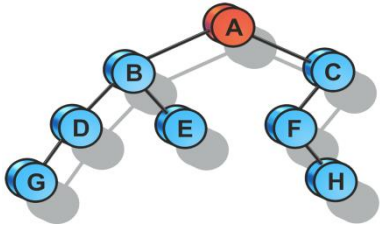


Алгоритмы программирования и структуры данных

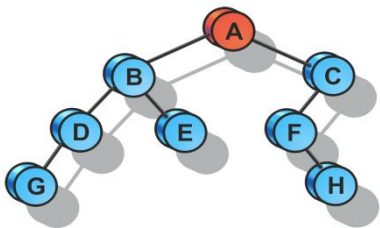
Поиск подстрок



Алгоритмы программирования и структуры данных

Поиск подстрок

Поиск подстрок 3. Алгоритм Кнута — Морриса — Пратта



Вспомним простой алгоритм

. **В** **А** **В** **С** **А** **А** **В** **А** **В** **С** . .

А В С А А В **Д** . . .

А В С А А В . . .

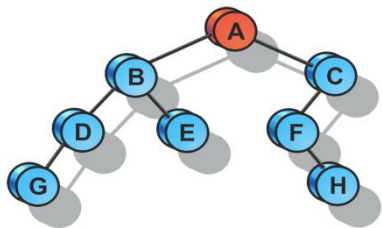
А В С А А . . .

А **В** С А . . .

А В **С** . . .

А В . . .

А . . .



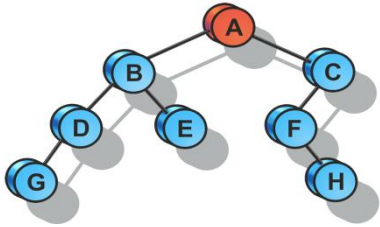
Большие сдвиги

. В **А В С А А В** А В С . .

А В С А А В **Д** . . .

А В **С** . . .

А . . .



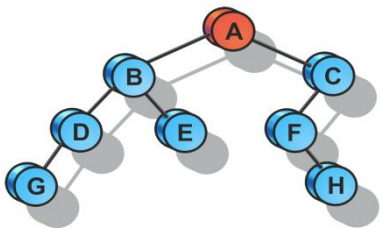
Префикс-функция

$\text{pref}(S)$ = максимальный по длине собственный префикс строки S , являющийся также ее суффиксом

$\text{pref}(\text{"ABABAB"}) = \text{"ABAB"}$

$\text{pref}(\text{"ABCAAB"}) = \text{"AB"}$

$\text{pref}(\text{"AB"}) = \text{" "}$



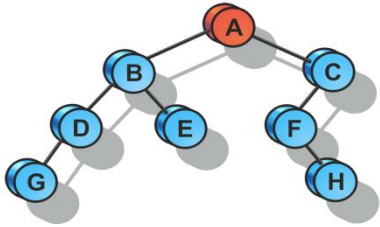
Большие сдвиги с использованием префикс-функции

. В **А В С А А В** А В С . .

А В С А А В **Д** . . .

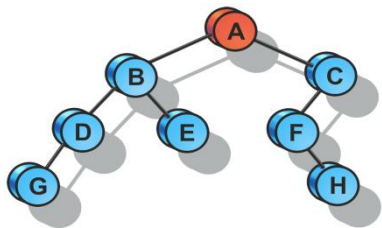
А В **С** . . .

А . . .



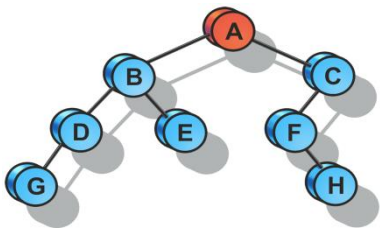
Префикс-функция для префиксов

- Посчитаем префикс-функцию для всех префиксов строки P.
- Префикс-функция от префикса – тоже префикс
- Достаточно хранить ее длину
- $p[i] = |pref(P[0..i-1])|$



Пример

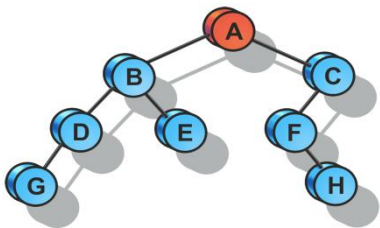
		А В А С А В А В А С В										
p	-	0	0	1	0	1	2	3	2	3	4	0
	p[0]											p[n]



Алгоритм поиска подстроки T

Состояние

						ⁱ							
S	.	B	A	B	C	A	A	B	A	B	C	.	.
T		A	B	C	A	A	B	.	.	.			
								_j					



Алгоритм поиска подстроки T

Случай 1. $S[i] == T[j]$

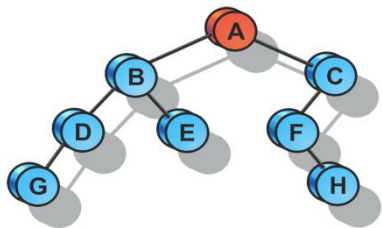
					i					
S	.	B	A	B	C	A	A	B	A	B
T		A	B	C	A	A	B	A	.	.
							j			

$i++$

$j++$

[illegible]

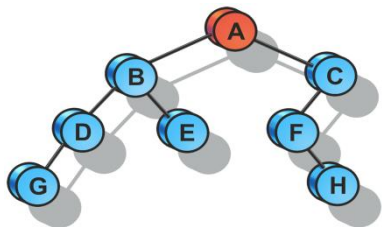
11



Алгоритм поиска подстроки T

Случай 3. $S[i] \neq T[j]$, $j=0$

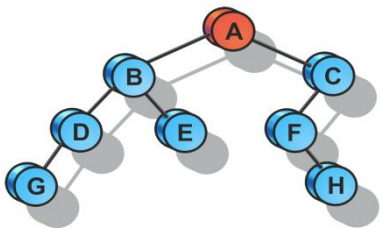
						ⁱ							
S	.	B	A	B	C	A	A	B	B	A	B	.	.
T		A	B	C	A	A	B	D	.	.	.		
			A	B	C	.	.	.					
				A	.	.	.						
					A	.	.	.					
						A	.	.			i++		
							_j						



Код

```
find(s, t: String):  
    i = 0, j = 0  
    while i < n && j < m:  
        if s[i] == t[j]:  
            i++, j++  
        else:  
            if j > 0:  
                j = p[j]  
            else:  
                i++  
    if j == m:  
        return i - m  
    else:  
        return -1
```

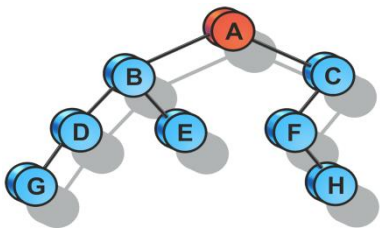
Время работы: $O(N + M)$



Алгоритм построения массива P

Состояние

						i			
T	.	B	A	B	C	A	A	B	A
T		A	B	C	A	A	B	.	.
							j		



Алгоритм построения массива P

Случай 1. $T[i] == T[j]$

					i								
T	.	B	A	B	C	A	A	B	A	B	C	.	.
T			A	B	C	A	A	B	A	.	.	.	
									j				

```

p[i+1] = j+1
i++
j++

```



Diagram illustrating a sequence alignment between two strings, T and T, with a highlighted mismatch at position j .

String 1 (Top): . **B** **A** **B** **C** **A** **A** **B** **A** **B** **C** . .

String 2 (Bottom): A B C A A B **D** A B C

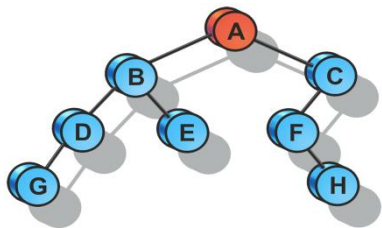
The alignment shows a mismatch at position j (indicated by the red box around 'D'). The characters 'A', 'B', and 'C' are aligned correctly in green. The characters 'B', 'A', 'A', 'B', 'A', and 'B' are bolded in the top string. The character 'D' is highlighted in a red box in the bottom string.

16



Случай 3. $T[i] \neq T[j]$, $j=0$

17

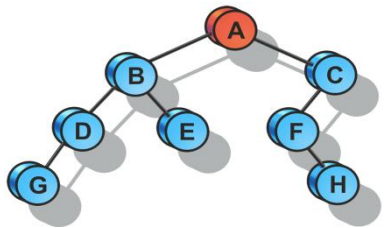


Алгоритм построения массива P

Пример

	i										
	A	B	A	C	A	B	A	B	A	C	B
	A	B	A	C	A	B	A	B	A	C	B
	j										
p	-	0	-	-	-	-	-	-	-	-	-

```
p[i+1] = 0
i++
```



Алгоритм построения массива P

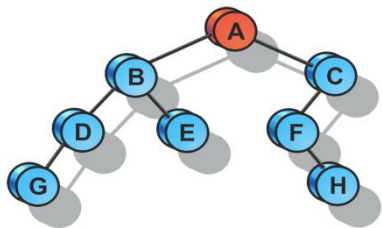
Пример

		i								
A	B	A	C	A	B	A	B	A	C	B
		A	B	A	C	A	B	A	B	A
		j								
p	-	0	0	-	-	-	-	-	-	-

```

p[i+1] = j+1
i++
j++

```

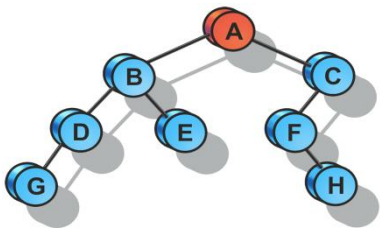


Алгоритм построения массива P

Пример

			i							
A	B	A	C	A	B	A	B	A	C	B
			j							
				A	B	A	C	A	B	A
p	-	0	0	1	-	-	-	-	-	-

$$j = p[j]$$

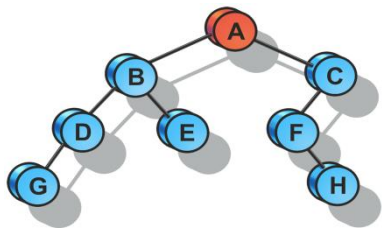


Алгоритм построения массива P

Пример

			i								
	A	B	A	S	A	B	A	B	A	S	B
			A	B	A	S	A	B	A	B	A
			j								
p	-	0	0	1	-	-	-	-	-	-	-

$p[i+1] = 0$
 $i++$



Алгоритм построения массива P

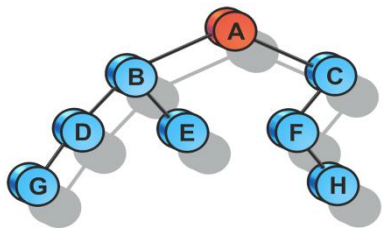
Пример

				ⁱ							
A	B	A	C	A	B	A	B	A	C	B	
				^A	B	A	C	A	B	A	B
				_j							
p	-	0	0	1	0	-	-	-	-	-	-

```

p[i+1] = j+1
i++
j++

```



Алгоритм построения массива P

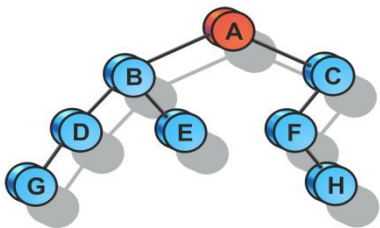
Пример

				ⁱ							
A	B	A	C	A	B	A	B	A	C	B	
				A	B	A	C	A	B	A	B
				_j							
p	-	0	0	1	0	1	-	-	-	-	-

```

p[i+1] = j+1
i++
j++

```



Алгоритм построения массива P

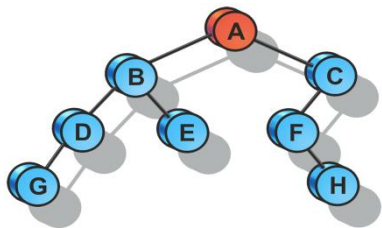
Пример

				i												
A	B	A	C	A	B	A	B	A	C	B						
					j											
						A	B	A	C	A	B	A	B	A	C	B
p	-	0	0	1	0	1	2	-	-	-	-	-	-	-	-	-

```

p[i+1] = j+1
i++
j++

```

Алгоритм построения массива P

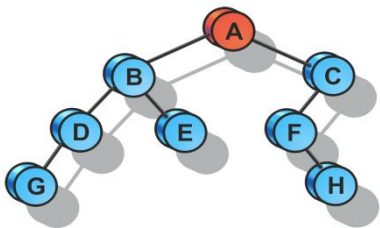
Пример

						ⁱ											
A	B	A	C	A	B	A	B	A	C	B							
						^j	A	B	A	C	A	B	A	B	A	C	B
p	-	0	0	1	0	1	2	3	-	-	-	-					

```

p[i+1] = j+1
i++
j++

```



Алгоритм построения массива P

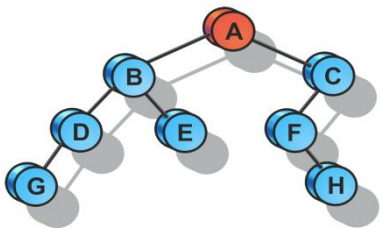
Пример

							ⁱ					
A	B	A	C	A	B	A	B	A	C	B		
							^j					
							A	B	A	C	A	B
p	-	0	0	1	0	1	2	3	2	-	-	-

```

p[i+1] = j+1
i++
j++

```

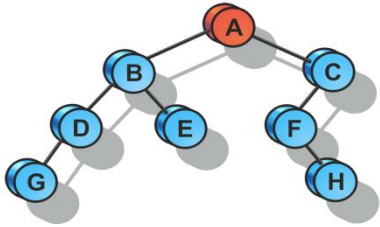



Алгоритм построения массива Р

Пример

[illegible]

```
p[i+1] = 0
i++
```

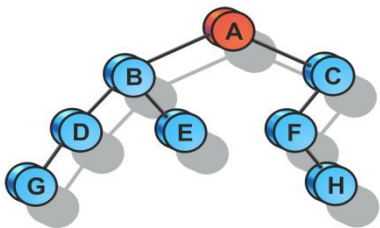


Алгоритм построения массива Р

Пример

$$\begin{array}{cccccccccccc}
 & & & & & & & & & i \\
 A & B & A & C & A & B & A & B & A & C & B \\
 & & & & & & & & & & A & B & A & C & A & B & A & B & A & C & B \\
 & & & & & & & & & & j \\
 p & - & 0 & 0 & 1 & 0 & 1 & 2 & 3 & 2 & 3 & 4 & 0
 \end{array}$$

```
p[i+1] = 0
i++
```



Код

```
buildP(t: String):  
    i = 1, j = 0  
    while i < n:  
        if t[i] == t[j]:  
            p[i + 1] = j + 1  
            i++, j++  
        else:  
            if j > 0:  
                j = p[j]  
            else:  
                p[i + 1] = 0  
            i++
```

Время работы: $O(N)$