Префикс-функция (также известная как Z-функция или функция наибольшего общего префикса) является эффективным алгоритмом для поиска всех вхождений заданной подстроки в строке. Алгоритм работает следующим образом:

1. Вычисляется префикс-функция для строки, в которой нужно искать подстроку.
2. Затем, строка конкатенируется с символом-разделителем и искомой подстрокой.
3. Вычисляется префикс-функция для результирующей строки.
4. Индексы, на которых значение префикс-функции равно длине искомой подстроки, являются местами вхождения подстроки в исходной строке.

В представленной реализации метода Init выделяется динамическая память для массива значений префикс-функции. Если вызвать Init повторно, то произойдет утечка памяти, так как предыдущий массив не будет освобожден. Для устранения этой проблемы необходимо освобождать память перед выделением новой. Это можно сделать в конструкторе, проверяя, был ли ранее выделен массив, и освобождая его перед новым выделением памяти.

В алгоритме вычисления префикс-функции значение для индекса i зависит от значения для индекса i - 1. Поэтому циклы начинаются с индекса 1, так как для индекса 0 значение префикс-функции всегда равно 0 (пустая строка является префиксом любой строки). Следовательно, последний индекс, для которого нужно вычислять значение префикс-функции, равен N - 1, где N - длина строки.

В алгоритме вычисления префикс-функции сравнения символов происходят внутри вспомогательного цикла while, который выполняется до тех пор, пока не будет найден наибольший общий префикс/суффикс. В худшем случае (когда строка состоит из одинаковых символов) этот цикл будет выполняться i раз для каждого индекса i. Следовательно, общее количество сравнений символов в худшем случае будет иметь сложность О(N2), где N - длина строки.

Результаты экспериментов показали, что количество сравнений символов растет линейно относительно длины строки, что соответствует теоретической оценке сложности O(N) для алгоритма вычисления префикс-функции. Увеличение размера алфавита приводит к небольшому увеличению количества сравнений символов, но не меняет общую линейную зависимость.

|  |  |
| --- | --- |
| N | Ср. знач количества сравнений |
| 10 | 1,4 |
| 30 | 7 |
| 50 | 12,6 |
| 70 | 17,6 |
| 100 | 23,6 |

В представленной реализации класса PF при копировании объектов или присваивании одного объекта другому возникнет проблема неявного копирования указателя pv, что приведет к ситуации, когда несколько объектов будут указывать на одну и ту же область динамической памяти. При уничтожении одного из объектов эта область памяти будет освобождена, и остальные объекты будут содержать некорректные указатели, что приведет к неопределенному поведению программы.

Для решения этой проблемы необходимо определить конструктор копирования и оператор присваивания, которые будут выполнять глубокое копирование динамически выделенного массива pv. В этом случае при создании нового объекта путем копирования или присваивания будет выделяться новая область памяти и копироваться содержимое исходного массива.