Условие: "В дальнейшем предполагается, что первым компонентом входа является регулярное выражение в обратной польской записи, задающее язык L. Условия задач:

17. Даны α и слово u ∈ (a; b; c)*. Найти длину самого длинного суффикса u, являющегося также суффиксом некоторого слова в L."

Состояния автоматов хранятся в массиве conditions структур Condition. Структура включает в себя четыре (по одному для каждой буквы, включая ε) массива, в каждом из которых хранится список номеров состояний, из которых можно попасть в данное по соответствующей букве (логичнее хранить состояния, в которые можно попасть по букве, но специфика конкретной задачи требует такой реализации). Каждому автомату соответствует класс NSM (Nondeterministic state machine). В нём хранятся номера начального и завершающего состояний.

В классе реализованы методы:

0) NSM (char c) - конструктор. В conditions дописываются два новых состояния. Создаёт автомат с двумя состояниями и переходом между ними.

1) Concatenation(NSM* second_part) - конкатенация автоматов. A.Concatenation(B): Добавляется ε-переход из завершающего состояния A в начальное состояние B. Новое начальное состояние - начальное состояние A, новое завершающее состояние - завершающее состояние B.

2) Disjunction(NSM* second_part) - дизъюнкция автоматов. A. Disjunction (B): Добавляются ε-переходы из начального состояния A в начальное состояние B и из завершающего состояния A в завершающее состояние B. Новое начальное состояние - начальное состояние A, новое завершающее состояние - завершающее состояние B.

3) Iteration() - итерация автомата ("навешивание" звезды Клини). Добавляются ε-переход из завершающего состояния в начальное, оно становится новым завершающим.

По полученному на входе регулярному выражению в обратной польской записи с помощью стека промежуточных автоматов строится НКА. Регулярное выражение разбирается по символам. Если символ - буква, то создаётся автомат с переходом по этой букве. Если операция - то оно производится с нужным количеством верхних в стеке автоматов, вместо которых в стек кладётся результат.

Поставленную в условии задачу решает рекурсивная функция Find_suff. Суть её работы заключается в том, чтобы проходить по автомату по инвертированным переходам пока путь удовлетворяет суффиксу данной строки.

Функция принимает строку s, текущее рассматриваемое состояние cur_cond и промежуточный результат cur_res. Изначально в качестве текущего состояния передаётся завершающее, в качестве текущего результата - 0.

Функция вызывает себя для каждого состояния, из которого текущее достижимо по ε-переходу, выбирая наилучший из возвращённых и текущего результатов. При этом она запоминает все совершённые подряд ε-переходы и, если состояние повторяется, заканчивает работу. Это позволяет избежать бесконечного блуждания по ε-циклу.

Затем функция вызывает себя для каждого состояния, из которого текущее достижимо по последнему символу строки, передавая в параметрах укороченную на 1 символ строку и увеличенный на 1 текущий результат, выбирая наилучший из возвращённых и текущего результатов.