Задача 1

Полностью объяснить задачу из второго задания № 5. Под «полностью объяснить» подразумевается все — от свойств используемых структур данных до строчек кода.

Формат сдачи — предоставленный исходный код плюс устная беседа по нему.

Задача 2

Разобрать алгоритм наивного поиска максимальной повторяющейся подстроки (см. ниже). Оценить асимптотическую сложность алгоритма.

Формат сдачи — названная асимптотическая сложность (как «О большое от чего-то») плюс устная беседа по алгоритму.

Задача 3

На базе любой удобной задачи второго задания реализовать структуру данных и интерфейс к ней (указаны ниже). Программа при запуске должна читать команды из файла commands.txt, выполнять их, печатать результат на экран и завершаться. Никаких интерактивных действий от пользователя не предполагается.

Структура данных: двухсвязный список

Тип данных: строка не более 15 символов

Состав инструкций в commands.txt:

INSERT A — добавить элемент со значением A, ничего не печатать. Если такой элемент уже есть — добавить еще одну копию.

FIND A — найти элемент со значением A. Если элемент найден, напечатать FOUND. Если не найден — напечатать NOT FOUND.

DELETE A — найти и удалить элемент со значением A. Если элемента нет — ничего не делать. Если элементов несколько — удалить одну из копий.

DEDUP A — из всех вхождений элемента со значением A оставить только одно (первое), остальные удалить.

Пример:

Инструкции в файле:

INSERT alice

INSERT alice

INSERT alice

DEDUP alice

DELETE alice

FIND alice

Вывод программы:

NOT FOUND (так как после DEDUP осталась только одна alice, которую потом удалили)

Формат сдачи — предоставленный исходный код плюс устная беседа по нему.

Наивный поиск максимальной повторяющейся подстроки

Строка x длины |x| = m записывается как $x_1x_2 \dots x_m$, где x_i представляет *i*-й символ *x*.

Подстрока $x_ix_{i+1} \dots x_j$ строки x, где *i<=j<=m*, будет обозначаться x(i,j). В случае, когда i>j, обращенная подстрока обозначается так $x_R(i,j)$.

Обычно x будет обозначать искомый образец, а y – текстовую строку; |x| = m, |y| = m и, конечно, m<=n.

Пример:

x = trismegistus
|x| = 12
x(7,10) = gist
$x_R(7,4)$ = gems

Задача нахождения самой длинной подстроки (longest repeated substring), появляющейся в данной строке более одного раза, можно сформулировать следующим образом.

Для данной строки y, |y| = n > 0, найти самую длинную подстроку, встречающуюся в y больше одного раза.
Самая длинная повторяющаяся подстрока – это самая длинная из строк максимальной длины x, такая что y = uxvxw для некоторых строк u, v и w, где |u| > 0, |v| > 0 и |w| > 0.

Максимальная повторяющаяся подстрока – это подстрока, имеющая в данной строке не меньше двух различных вхождений, причем сопоставление нельзя продолжить ни в одном направлении. Рассмотрим, например, строку PABCQRABCSABTU. Подстроки A, B, C, AB, BC и ABC повторяются. Строки AB и ABC являются максимальными повторяющимися подстроками, и, таким образом, ABC является самой длинной повторяющейся подстрокой.

Наивный подход к решению этой задачи состоит в следующем. Строится матрица M размерности n * n, такая что $M_{i,j}$ = 1 если $y_i = y_j$, и $M_{i,j}$ = 0 в противном случае. За исключением главной диагонали, все диагонали в матрице, состоящие из идущих подряд `1', представляют максимальные повторяющиеся подстроки, самая длинная из которых является, таким образом, самой длинной повторяющейся подстрокой. Обратите внимание, что матрица симметрична относительно главной диагонали, поэтому достаточно вычислять и обрабатывать только одну, скажем, верхнюю, ее половину (то есть элементы $M_{1\dots n-1,i+1\dots n+1}$).

Наивный подход к задаче отыскания самой длинной повторяющейся подстроки для текстовой строки y может быть основан на матрице совпадений M размерности n * n. Элементы этой матрицы определяются следующим образом:

$$M_{i,j} = \begin{cases} 1, & \text{если } y_i = y_j \\ 0 & \text{в остальных случаях} \end{cases}$$

Такая матрица симметрична относительно главной диагонали. Поэтому надо вычислить только элементы над ней или под ней. Диагонали из смежных ‘1’, за исключением главной, представляют повторения в строке y, там-то и нужно искать максимальные повторяющиеся подстроки. Вот пример матрицы совпадений для строки PABCQRABCSABTU

	j	1	2	3	4	5	6	7	8	9	10	11	12	13	14
i		**P**	**A**	**B**	**C**	**Q**	**R**	**A**	**B**	**C**	**S**	**A**	**B**	**T**	**U**
1	**P**	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	**A**	0	1	0	0	0	0	1	0	0	0	1	0	0	0
3	**B**	0	0	1	0	0	0	0	1	0	0	0	1	0	0
4	**C**	0	0	0	1	0	0	0	0	1	0	0	0	0	0
5	**Q**	0	0	0	0	1	0	0	0	0	0	0	0	0	0
6	**R**	0	0	0	0	0	1	0	0	0	0	0	0	0	0
7	**A**	0	1	0	0	0	0	1	0	0	0	1	0	0	0
8	**B**	0	0	1	0	0	0	0	1	0	0	0	1	0	0
9	**C**	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10	**S**	0	0	0	0	0	0	0	0	0	1	0	0	0	0
11	**A**	0	1	0	0	0	0	1	0	0	0	1	0	0	0
12	**B**	0	0	1	0	0	0	0	1	0	0	0	1	0	0
13	**T**	0	0	0	0	0	0	0	0	0	0	0	0	1	0
14	**U**	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Из приведенной выше матрицы можно видеть, что максимальными повторяющимися подстроками в данном примере являются ABC, с вхождениями y(2, 4) и y(7, 9), и AB, с вхождениями y(2, 3), y(7, 8) и y(11, 12). Легко выбрать более длинную из них, а именно, ABC.