Семинар 9 октября

1. Даны последовательности чисел. Каждая последовательность заканчивается числом 0. Надо загрузить эти последовательности в память, отсортировать каждую из них, а затем вывести эти отсортированные последовательности в «почти лексикографическом» порядке. Но только если одна последовательность короче другой и точно совпадает с началом другой, то сначала должна идти длинная, а не короткая.

2. Даны строки разной длины, состоящие только из цифр. Нужно загрузить строки в память и отсотрировать в числовом порядке. (Примечание: преобразовывать строки в числа с помощью функций `atoi` и аналогичных нельзя.)

3. Напишите функцию `template <typename T> void printBinary(T number)`, которая бы выводила на экран битовое представление числа. Протестируйте корректность ее работы для типов `char`, `int`, `long`.

4. Напишите функцию поиска подстроки в строке. Совпадением считается подстрока, совпадающая с образцом с точностью до перестановки букв. Строка может содержать только ASCII-символы, сложность решения должна быть $O(Cn)$, где n – длина строки, C – длина алфавита.

5. N-граммой строки называется ее подстрока длины n. Напишите функцию
`void calcNGramms(const string& str, size_t n, map<string, size_t>* result)`,
которая бы считала n-граммы по заданной строке и записывала бы в `*result` все n-граммы текста с указанием того, сколько раз такая n-грамма встретилась.

6. Реализуйте класс `Queue` на основе двух векторов.

7. Реализуйте класс `Stack` на основе вектора.