Факультет компьютерных технологий и прикладной математики

Кафедра вычислительных технологий

02.03.02

Информационная безопасность

Лабораторная работа № 2

Тема: проектирование алгоритмов поддержки информационной безопасности

Цель работы: научиться проектировать алгоритмы поддержки информационной безопасности.

Задание: необходимо разработать алгоритмы и программы решения задач 0-5 на языке C++.

Указания к работе. На занятии студенты решают задачу № 0. Далее каждому студенту выдается своя задача. Номер задачи, которую решает студент, вычисляется по формуле $N = (X\ mod\ 5) + 1$, где X – номер студента в списке. Студент разрабатывает алгоритм и программу решения задачи на языке C++ на лабораторном занятии. Если студент не успел в течение занятия разработать программу, то ему необходимо закончить разработку к следующему лабораторному занятию, используя систему управления версиями и разместив её на личный Git репозиторий. За работу на занятии студент может получить оценку «зачтено», «удовлетворительно», «хорошо», «отлично». Если студент не получил оценку, то ему будет необходимо подготовить отчёт по задаче и реализовать программу, используя систему управления версиями Git и размещая её репозиторий на GitHub.

0. RLE. Компрессия RLE – это способ сжатия данных, в которых имеется много подряд идущих символов. При этом более одного вхождения символа заменяется на пару (количество вхождений - символ). В входном файле input.txt задан текст, состоящий из букв латинского алфавита. Примените к нему с помощью написанной программы преобразование RLE, т.е. замените N ($N > 1$) вхождений некоторого символа X, которые подряд идут в строке на NX. Длина

входной строки не более 10000 символов. Преобразованный текст программа должна поместить в текстовый файл output.txt. Пример:

input.txt	output.txt
BCCAAAAAAAAAAABBBD	B2C11A3BD

1. Кодовый замок. Есть кодовый замок. На нём N переключателей ($1 \leq N \leq 26$), промаркированных различными буквами. Каждый переключатель может быть или в положении ON, или в положении OFF. При установке замка владелец ставит в соответствие каждой из N букв некоторый код – натуральное число. Различные буквы имеют различные коды C_1, C_2, …, C_N ($1 \leq C_j \leq 64000, 1 \leq j \leq N$). Для того, чтобы открыть замок, владелец устанавливает некоторые переключатели в положение ON, а оставшиеся – в положение OFF. Замок открывается, когда сумма всех чисел-кодов, соответствующих выключателям, находящимся в положении ON, делится без остатка на N, т.е. существует некоторое подмножество кодов $C_{j_1}, C_{j_2}, \ldots, C_{j_k}$ ($k \leq N$), сумма всех членов которого делится на N. Взломщик не знает значений кодов и пытается открыть замок, перебирая некоторые комбинации букв и устанавливая соответствующие переключатели в положение ON. Требуется составить программу, которая должна определить, возможно ли открыть замок, т.е. существует ли такое подмножество букв, что сумма кодов, соответствующих этим буквам, делится без остатка на N (возможные ответы программы: YES или NO). В случае, если это возможно, программа должна выдать какое-то подмножество кодов $C_{j_1}, C_{j_2}, \ldots, C_{j_k}$ и P – количество всех возможных комбинаций кодового замка, которые его открывают. Входные данные задаются в текстовом файле input.txt, а результаты – в текстовом файле output.txt. Формат входного файла input.txt:

N

$C_1\ C_2\ \ldots, C_N$

Формат выходного файла output.txt:

YES/NO

$C_{j_1}\ C_{j_2}\ \ldots\ C_{j_k}$

P

Например, input.txt:

4

15 2 6 19

Тогда output.txt:

YES

2 6

16

2. Фокусник. Старый фокусник решил передать секрет одного фокуса своему ученику. Фокус заключается в следующем: зритель угадывает натуральное число от 100 до 10^{200}, затем вычитает из числа сумму его цифр. В полученном числе он зачёркивает любую ненулевую цифру и называет все оставшиеся цифры фокуснику. Выслушав названные цифры, фокусник, погодя, называет цифру, которая была зачёркнута. Ученик не силён в вычислениях. Поэтому требуется написать программу, которая, получив на вход названные цифры, выдаёт зачёркнутую зрителем цифру. Вход программы представлен текстовым файлом input.txt, в котором в одну строку без пробелов заданы называемые зрителем цифры. Выход программы представлен текстовым файлом output.txt, в который записывается зачёркнутая цифра. Пример:

input.txt	output.txt
12345674	4

3. Тайский шифр. Когда-то король королевства Хай-Тай решил придумать шифр для кодирования секретных посланий своему послу в далёкой стране. Алфавит в Хай-Тае состоит из большого числа иероглифов. Все они нумеруются числами от 1 до N ($1 \leq N \leq 30000$). Исходный текст (набор иероглифов без пробелов или знаков пунктуации) преобразуется в последовательность чисел $a_1, a_2, \ldots, a_m$ ($1 \leq m \leq 30000$) путём замены каждого иероглифа его номером. Кодировщик шифрует эту последовательность умножением каждого номера на константу D с последующим увеличением на константу C ($0 \leq C \leq 30000, 0 \leq D \leq 30000$). Если результат i больше, чем N, то он (т.е. i) должен быть заменён i-ым номером в последовательности 1, 2, …, N, 1,2, …, N, 1, 2, …, N, … . Например, результат $i = 35$ заменяется на 5, если $N = 15$. Поэтому результаты шифруются последовательностью чисел $b_1, b_2, \ldots, b_m$ из диапазона $[1; N]$. Заменив число b_k иероглифом с таким номером в алфавите, кодировщик получит зашифрованный текст. Посол, получив зашифрованный текст, хочет расшифровать сообщение. Во избежание международных осложнений требуется написать программу, которая должна определить, может ли посол однозначно расшифровать сообщение. Программа должна выдать вердикт YES или NO. Если программа отвечает YES (т.е. сообщение однозначно расшифровывается), то она выдаёт и расшифрованное сообщение $a_1, a_2, \ldots, a_m$. Входные данные задаются в текстовом файле input.txt. В первой строке этого файла задаются натуральные числа N и m, разделённые одним или несколькими пробелами. Во второй строке задаются целые числа D и C, разделённые одним или несколькими пробелами. В третьей строке записаны целые $b_1, b_2, \ldots, b_m$, разделённые пробелами. Результат помещается в текстовый файл output.txt. В первой его строке пишется слово YES или NO. В случае

YES вторая строка содержит расшифрованную последовательность $a_1, a_2, \ldots, a_m$ чисел, разделённых пробелами. Пример:

input.txt	output.txt
13 7 7 3 5 4 9 2 7 8 1	YES 4 2 12 11 8 10 9

4. Super RLE. Можно пойти и дальше (см. задачу № 0), попытавшись написать программу, реализующую преобразование Super RLE. Для этого повторяющийся N раз фрагмент F исходного текста заменить на $N(F)$. Например, текст AAAA заменяется на 4(A), текст ABCBCD заменяется на A2(BC)D, а текст AAABBAAABBCCCCD заменяется на 2(3(A)2(B))4(C)D. Сжатый в такой форме текст выводится в файл output.txt. Если возможно несколько вариантов свёртки, то в output.txt помещается свёртка по возможности наименьшей длины.
5. Шпионский сленг. Секретные сообщения между агентами кодируются 25-языком. В этом языке используется 25-алфавит, который совпадает с латинским алфавитом, но в нём отсутствует буква Z, т.е. 25-алфавит содержит 25 латинских букв от A до Y в том же порядке, что и латинский алфавит. Каждое слово такого языка состоит ровно из 25 различных букв. Слово записывается в таблице размера 5×5 строка за строкой. Например, слово ADJPTBEKQUCGLRVFINSWHMOXY будет записано так:

   A D J P T
   B E K Q U
   C G L R V
   F I N S W
   H M O X Y

   Правильным словом 25-языка считается такое слово, буквы в котором (если записать его в виде 5×5-таблицы) расположены по

возрастанию их номера в алфавите в каждой строке и в каждом столбце. Поэтому слово ADJPTBEKQUCGLRVFINSWHMOXY является правильным, а слово ADJPTBEGQUCKLRVFINSWHMOXY – нет (возрастающий порядок нарушается во втором и третьем столбцах). Лексикон агента состоит из всех правильных слов 25-языка. Слова расположены в возрастающем лексикографическом порядке (как это принято в словарях) и пронумерованы, начиная с 1. Например, слово ABCDEFGHIJKLMNOPQRSTUVWXY имеет номер 1, а слово ABCDEFGHIJKLMNOPQRSUTVWXY имеет номер 2. Во втором слове, по сравнению с первым, буквы U и T поменялись местами. Для упрощения нелёгкой жизни агента требуется написать программу, которая по заданному порядковому номеру слова определяет само это слово. В лексиконе агента не более 2^{31} слов. Программа на входе получает порядковый номер некоторого правильного слова 25-языка. Единственная строка на выходе – это слово с заданным порядковым номером. Например, на входе 2, на выходе ABCDEFGHIJKLMNOPQRSUTVWXY.