Информатика

Лабораторная работа №2
Поразрядные и логические операции

Задание: написать программу, которая получает случайное целое типа int, выводит его двоичное представление на экран (необходимо выводить все разряды числа), выполняет преобразование в соответствии с вариантом (Табл. 1), затем выводит на экран двоичное представление результата преобразования.

Программу необходимо реализовать в двух вариантах:

1. На языке C.
2. На ассемблере для x86-64.

В обоих случаях в качестве целевой платформы необходимо использовать ОС Linux, в качестве ассемблера для x86-64 использовать NASM, а компилировать программу на C с помощью gcc или clang.

После написания программ на C и на ассемблере необходимо выполнить дизассемблирование программы на C и сравнить получившийся код с ассемблерным кодом, который был написан для решения задачи вручную.

Отчет должен содержать:

- правильно оформленную титульную страницу;
- задание;
- блок-схему алгоритма преобразования для программы на C;
- текст программы на C с комментариями;
- текст программы на ассемблере NASM с комментариями;
- дизассемблерный листинг существенных частей программы на C с добавленными комментариями или пояснениями;
- краткий анализ по результатам сравнения программы на ассемблере и дизассемблированной программы на C;
- скриншоты прогонов программ на различных исходных данных.

Замечание 1. В результате преобразования должно измениться исходное число (переменная), недостаточно просто вывести результат преобразования на экран.

Замечание 2. Если операция предусматривает использование одного или нескольких случайных параметров (например, сдвиг на случайное число битов), их значения следует выводить на экран.

Замечание 3. Преобразование должно осуществляться **только** с помощью битовых операций (сдвиги, логические операции). Использование стандартных массивов и VLA, битовых полей структур и объединений или контейнеров STL для представления битов и байтов числа в программе на C **запрещено!**

Замечание 4. В программе на C использование ассемблерных вставок запрещается!

Замечание 5. В дизассемблерный листинг, приводимый в отчете, следует включить функцию main() и дополнительные определенные в программе функции, участвующие в преобразовании числа (если есть).

Таблица 1. Варианты заданий

№ варианта	Задание	Пример
1	Изменить порядок следования битов в числе на обратный.	11010011 → 11001011 *
2	Изменить порядок следования нечетных битов в числе на обратный.	11110000 → 01011010 *
3	Сдвинуть биты в каждом байте циклически вправо на случайное число N из диапазона 0..7.	0xDEADBEEF → 0xB76BAFFB (N = 2)
4	Если в числе встречается последовательность битов 11011, то заменить её на 01110. В преобразованном числе исходная последовательность встречаться не должна.	0xDEADBEEF → 0x5CACBAAE
5	Каждую младшую тетраду каждого байта сдвинуть циклически влево на число, содержащееся в двух старших битах старшей тетрады.	11010011 → 11011001 *
6	Старшую тетраду каждого байта числа заменить результатом операции «стрелка Пирса» старшей и младшей тетрад, а младшую тетраду – результатом операции «штрих Шеффера» старшей и младшей тетрад исходного байта.	11010011 → 00001110 *
7	Если в числе встречается последовательность битов 000, заменить её на 0110 (лишние разряды сдвигать влево). В преобразованном числе исходная последовательность встречаться не должна.	11010001 → 10101101 *
8	Старшую тетраду в нечетных байтах числа заменить результатом операции «исключающее ИЛИ» старшей и младшей тетрад, а младшую тетраду в четных байтах – побитовым отрицанием результата операции «исключающее ИЛИ» старшей и младшей тетрад.	0xDEADBEEF → 0x3EA85EEE
9	Назовем сверткой байта порядка N операцию циклического сдвига старшей тетрады на N битов вправо, а младшей тетрады на N битов влево. Выполнить свертку всех байтов на случайное число из диапазона 0..3.	0xDEADBEEF → 0x7BA7EBBF (N = 2)
10	Назовем рангом байта значение трех его старших битов. Изменить порядок следования байтов числа по возрастанию их рангов.	0xDEADBEEF → 0xEFDEADBE **
11	Назовем характеристикой байта количество единичных битов. Упорядочить байты числа по возрастанию их характеристик.	0xDEADBEEF → 0xEFDEBEAD **
12	Назовем триплетом группу из трех битов. Если соседние триплеты одинаковы, инвертировать в младшем старший бит, а в старшем триплете – младший.	0xDEADBEEF → 0xDEAD7EEF
13	В четных байтах переместить нулевые биты в старшие биты, а в нечетных байтах – в младшие.	0xDEADBEEF → 0xFC1FFC7F

14	Назовем триплетом группу из трех битов. В каждом третьем триплете, начиная с младшего, изменить порядок следования битов на обратный (2,5,8,…).	0xDEADBEEF → 0xDBAF3FAF
15	Назовем симметричным байт, в котором нулевой бит имеет такое же значение, что и седьмой, а первый – такое же, что и шестой. Изменить порядок следования симметричных байтов в числе на обратный.	0xDEADBEEF → 0xDEEFBEAD
16	Расположить тетрады числа в порядке возрастания.	0xDEADBEEF → 0xFEEEDDBA **
17	Назовем сверткой байта его арифметический сдвиг вправо на количество содержащихся в нем нулевых битов. Выполнить свертку всех байтов числа.	0xDEADBEEF → 0x37152F77
18	Назовем характеристикой байта разность между значениями младшей и старшей тетрад. Расположить байты числа в порядке убывания их характеристик.	0xDEADBEEF → 0xADBEDEEF **
19	Назовем симметричным байт, в котором значения разрядов младшей тетрады обратны значениям разрядов старшей тетрады. Если в числе есть симметричные байты, сделать их несимметричными.	0xDEADBEE1 → 0xDEADBEE2
20	Случайным образом изменить порядок следования тетрад в числе.	0xDEADBEEF → 0xEDDEFABE
21	Назовем серединой байта его биты 2..5. Выполнить циклический свиг середины нечетных по порядку следования байтов влево, а четных – вправо на случайное число из диапазона 0..3.	1010110111 → 1010011111 * (N = 2)
22	Выполнить операцию исключающее или (XOR) самой большой по значению тетрады в числе со всеми тетрадами, содержащими четное число.	0xDEADBEEF → 0xD15DB11F
23	Назовем дублетом группу из двух битов. Расположить дублеты числа в порядке возрастания.	101101001101 → 111110010100 **
24	Сдвинуть биты в каждом байте циклически вправо на число, содержащееся в двух старших битах байта.	0xDEADBEEF → 0xDB6DAFFD
25	Циклически сдвинуть нечетные биты в числе влево, а четные – вправо на случайное число из диапазона 0..3.	10110101 → 11000111 * (N=2)
26	Случайным образом изменить порядок следования битов в каждом байте числа.	0xDEADBEEF → 0x7B9EE7BF
27	Найти в числе непрерывную последовательность из 10 битов, имеющую наименьшее значение, и изменить в ней порядок следования битов на обратный.	0xDEADBEEF → 0xDFB53EEF
28	Назовем характеристикой байта количество нулевых битов. Упорядочить байты числа по убыванию их характеристик.	0xDEADBEEF → 0xEFDEBEAD **
29	Назовем рангом байта результат побитовой импликации младшей и старшей тетрад. Упорядочить байты числа по возрастанию их рангов.	0xDEADBEEF → 0xEFDEBEAD **
30	Назовем дублетом группу из двух битов. Заменить наиболее часто встречающийся дублет в числе на его инвертированное значение.	0xDEADBEEF → 0x12A18220

* – в данных примерах показано преобразование одного байта числа
** – когда речь идет о расположении битов/тетрад/… в порядке возрастания/убывания, их надо расположить справа налево (от младших битов к старшим)

Версия 1.2 от 27.10.2020