**4)**Разберите дизассемблерный листинг функции countLeadingZeros, уделите особое внимание ассемблерным инструкциям, с помощью которых реализованы арифметические действия, битовые операции, сравнения и логические операторы языка С.

Суть: проходим от младших битов к старшим. Если встречаем на пути 0, ничего не происходит. Если встречаем единицу, обнуляем этот бит и все младшие биты. В итоге маска будет состоять из единицы, стоящей на той же позиции, что и ведущая единица в нашем числе. Слева от этой единицы будут единички, справа-нули. Остаётся посчитать количество единиц. Это кол-во минус 1-искомое число.

0x08000474 F0030001 AND r0,r3,#0x01

Опять выполняем побитовое И для нашего аргумента и двойки в степени 0.

0x08000478 9901 LDR r1,[sp,#0x04]

Загружаем в регистр «единички»-число состоящее из 32-ух двоичных единиц

0x0800047A 1A08 SUBS r0,r1,r0

Вычитываем из «единичек» бит, который только что проверили. То есть если бит, который мы проверили, равен 1, то в «маске» (переменная number) образуется нуль на данной позиции и на всех позициях правее (в сторону младших битов).

Это выполнилась первая «скобочка» вычисления числа number.

0x0800047C 0FC9 LSRS r1,r1,#31

Логически сдвигаем «единички» на 31 бит вправо. Получаем 1.

0x0800047E F3C30440 UBFX r4,r3,#1,#1

Записываем в регистр 1-ый бит нашего числа

0x08000484 FB042111 MLS r1,r4,r1,r2

Из целой маски (r2) вычитаем произведение вида ((n&1<<1)==(1<<1))\*(numerals>>31)

0x08000488 4008 ANDS r0,r0,r1

Производим побитовое И между маской в этой итерации и итоговой маской.

Далее проделываются аналогичные операции для каждых скобочек.

С помощью функции из предыдущего упражнения подсчитывается количество единиц в маске. Возвращается в итоге полученное количество -1.