Problem sm05-2: c/floats/uint-fit-1

На стандартном потоке ввода задается последовательность 32-битных беззнаковых целых чисел. Последовательность заканчивается с концом файла.

Для каждого введенного числа на отдельной строке напечатайте 1, если это число может быть точно представлено в виде значения типа `float`, и 0 в противном случае.

В решении не используйте операции с вещественными числами.

Операцию проверки точной представимости числа выделите в отдельную функцию.

Problem sm05-3: c/floats/fpclass-1

Дан следующий перечислимый тип:

```
typedef enum FPClass
{
    FFP_ZERO, FFP_DENORMALIZED, FFP_NORMALIZED, FFP_INF, FFP_NAN
} FPClass;
```

Напишите следующую функцию:

```
FPClass fpclassf(float value, int *psign);
```

Функция на вход принимает число `value` типа `float` и классифицирует его. Если число не равно `NAN`, то в переменную `psign` возвращается знак числа (1 для отрицательных чисел). Для чисел `NAN` знак всегда должен быть 0.
При сдаче функции на проверку должно присутствовать только тело функции. Ни определение перечислимого типа `FPClass`, ни функция `main` присутствовать не должна.

Запрещается использовать стандартные функции и расширения gcc для классификации чисел.

Problem ku02-1: kr02-1 (upsolving)

На стандартном потоке ввода вводится последовательность описаний множеств целых чисел в интервале [1;32].

Каждое множество описывается возможно пустой последовательностью целых чисел в интервале [1;32] - чисел, принадлежащих множеству. Числа могут идти в произвольном порядке и могут повторяться. Описание множества заканчивается числом 0.

Найдите поэлементное равенство последовательности множеств, заданных на стандартном потоке ввода. То есть, если, например, задано три множества {1, 2, 3},

{3, 4}, {1, 5}, то сначала находится поэлементное равенство первых двух множеств, получается {3, 5 ... 32}, затем находится поэлементное равенство результата и третьего множества, получается {2, 4, 5}. Элемент считается принадлежащим поэлементному равенству двух множеств, если он либо не принадлежит ни одному из множеств, либо принадлежит сразу двум множествам.

Если во входном потоке подается только одно множество, результат работы совпадает с этим множеством. Поэлементное равенство двух пустых множеств дает полное множество.

Результат вычислений напечатайте на стандартном потоке вывода в виде 32-битного числа в шестнадцатеричном виде, где младшему биту соответствует значение 1, а старшему биту - значение 32.

В программе не должны использоваться конструкции, приводящие к Undefined Behavior.

Ответ выводите точно в том виде, как показано на примере, то есть без ведущих нулей и со строчными латинскими буквами 'a'-'f'.

Не забывайте выводить \n в конце вывода.

Examples

```
3 2 1 0 4 3 0 5 1 0
```

```
1a
```

Problem ku02-2: kr02-2 (upsolving)

Напишите функцию

```
int mystrdigcmp(const char *str1, const char *str2);
```

Функция должна лексикографически сравнивать строки, которые получались бы из строк `str1` и `str2`полным удалением символов десятичных цифр '0' - '9'. Функция возвращает значение 0, если две строки лексикографически равны, **положительное** значение, если первая строка лексикографически меньше второй, **отрицательное** значение, если первая строка лексикографически больше второй.
Гарантируется, что параметры `str1` и `str2` не равны `NULL`.
Запрещается использовать дополнительную память для преобразования строк. Запрещается использовать стандартные функции (кроме функций из ctype.h) и операции индексирования. Запрещается использовать операциюю `*(p + n)`. Используйте только указатели и указательную арифметику.

Например:

```
    mystrdigcmp("90102abs",  "512x")  ==  1  //  допустимо  любое
положительное значение
```