**Зад. D?. ДЕЛИМОСТ**

Петя вече е състезател по програмиране и старателно изучава двоичната бройна система. В една от почивките, след като се справила с подкарването на не лек алгоритъм, тя се вгледала в записките си и с учудване забелязала, че ако разгледа двоичното представяне на числото 8 – 1000(2) – като число в десетична бройна система, ще се окаже, че 8 дели без остатък 1000(10). Пробвала следващото число, но се оказало че при 9 не е така – 9 не дели 1001(10). Решила, че случаят с 8 е някакво изключение, но се загледала в числото 10. И останала изненадана! Оказало се числото 10 дели 1010(10). Същото се случило и при 16, тъй като 16 дели без остатък 10000(10). Ама че работа, помислила си Петя. Колко ли такива числа има? Искало й се да направи сама това изследване, но подготовката й не била достатъчна. Помогнете й. Напишете програма **delimost**, която по зададени две цели числа *A* и *B*, в десетичния запис на които има само цифрите 1 и 0, проверява за кои десетични числа *X* от интервала [*A*,*B*], също съставени само от цифрите 1 и 0, е изпълнено, че ако *Y* e десетичното число, чийто двоичен запис е *X*, то *Y* дели *X* без остатък.

На стандартния вход ще бъдат зададени двете цели числа *A* и *B*, 0 < *A* < *B* < 1020.

На отделни редове на стандартния изход програмата трябва да изведе всички десетични числа *Y*, за които двоичното им представяне *X*, разгледано като десетично число е в интервала [*A*,*B*] и *X* дели *Y.* Числата трябва да бъдат изведени в нарастващ ред. Ако в зададения интервал няма нито едно такова число, програмата трябва да изведе –1.

**ПРИМЕР**

**Вход Изход**

**1000 10000 8**

 **10**

 **16**

**РЕШЕНИЕ**

Решението на задачата е композиция от внимателно комбинирани познати техники. На първо място това е организирането на цикъла, който обхожда всички числа в зададения интервал, които съдържат в десетичното си представяне само 0 и 1. Очевидно е, че при стойности близки до зададените ограничания, това не може да стане с разглеждане на всички числа в интервала и проверяване кои от тях са от зададения вид. Ефективното решение е да се представи началото на интервала *M* в масив от тип char или в string и да се процедира както при работа с големи цели числа. За целта трябва да се имплементират само две функции – „добавяне на единица“ и „сравняване“, за да може да се установи достигането на края на интервала. При това добавянето на единица не трябва да се прави с общия алгоритъм за събиране на големи цели, а по специфичния за добавяне на единица алгоритъм: ако последната цифра на числото е 0 – заменяме я с единица; в противен случай заменяме всички единици в младшите разряди с нули и когато стигнем до първата нула (или първата позиция след най-старшия разряд) – заменяме я с единица. В авторовото решение тези две операции са реализирани с функциите void plusone() и bool compare().

След като организацията на основния цикъл е завършена остава само да пресметнем двете числа, които съответстват на низа от нули и единици – един път разгледан като десетично число *d* и втори път, като двоично число *b* – и да проверим дали *b* дели *d*. Двете пресмятания е най добре да се извършат с алгоритъма на Хорнер, имплементиран във функцията long long Horner(…). Пресмятанията по алгоритъма на Хорнер за големите тестове трябва непременно да се направят в променлива от тип unsigned long long, заради екстремалните стойности на *A* и *В*.

Красимир Манев