**АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА**

**СУМА НА ДВЕ ЧИСЛА**

Ако липсват *F* и *P*, задачата е тривиална: сума на „дълги“ неотрицателни цели числа, записани в бройна система с основа *b*. Наличието само на частта *F* в някое от събираемите (или в двете) не я променя съществено – достатъчно е, например, да допълним по-късото *F* с нули (или да симулираме това) и задачата се свежда до предишния прост вид. Остава да се занимаем със случая, в който поне в едно от събираемите присъства периодът *P*. Всъщност, винаги можем да считаме, че и двете събираеми имат части *P*, тъй като при липса можем да „долепим отдясно“ точка (ако липсва) и низ (0).

Прилагаме следния алгоритъм:

1. При въвеждането игнорираме водещите нули.
2. Ако в някое събираемо липсва частта *P*, полагаме *P* = (0).
3. Ако в някое събираемо липсва частта *F*, ползваме периода му, за да има в *F* поне един символ: изваждаме първия символ от периода пред периода и го преписваме на последно място в периода.
4. Ако дължините на частите *F* са различни, изравняваме ги, като (отново) ползваме периода на по-късата от тях (както в т. 3).
5. Получаваме периода *P* на сумата, като сумираме периодите на събираемите, започвайки от най-десните им символи. Указателите към *b*-ичните цифри, които се сумират, се въртят циклично, докато едновременно не посочат всеки първия символ на съответния период. Ако процесът завърши с пренос, увеличаваме получената сума (периода на резултата) с 1. Получения пренос подаваме **и** към следващата стъпка.
6. Получаваме частите *F* и *I* на сумата, като сумираме частите *F* и *I* на събираемите (вземаме предвид преноса от предишната стъпка). Това е стандартно събиране, тъй като сме изравнили частите *F* на събираемите. При получен пренос, частта *I* на резултата се увеличава с една цифра (и първата цифра, разбира се, става 1).
7. На този етап сме получили коректна сума, но не непременно нормална.
8. Нормализираме получения резултат:

* определя се минималният период;
* при получен едноцифрен период, равен по стойност на *b*-1, той се премахва, като частта *F* се увеличава с 1, а при нов получен пренос – и частта *I* се увеличава с 1; ако пък стойността му е 0, просто го премахваме.
* ако частта *P* е налична, евентуално се намалява частта *F*, чрез преместване на периода в частта *F*, докато последната цифра на *F* съвпада с последната цифра на *P* (обратно на изравняването в т. 4);
* ако периодът липсва, премахват се завършващите нули от частта *F*;
* ако периодът липсва и дължината на *F* стане нула, премахва се разделителната точка от записа на резултата.

Процесът на нормализиране, разбира се, е най-интересната част от задачата и най-съществената му съставка е намирането на минималния период в получената сума. Ако означим дължините на периодите на двете събираеми съответно с *m*1 и *m*2, полученият по описания алгоритъм низ-период има дължина *m*=*m*1*m*2. Можем да използваме различни алгоритми за намиране на търсената дължина, но бива да съобразим някои неща.

* Вероятностно погледнато, периодът е по-често голям, отколкото малък; по-често е *m*, отколкото някой негов делител. Затова, макар да е изкушаващо да започнем от малките делители на *m*, та първият намерен период да е и резултатът, по-изгодно е да започнем от най-големия му делител.
* С едно минаване по низа (а всъщност, може и по време на получаването му) можем:
* да проверим дали всички символи са еднакви;
* да използваме подходяща хеш-функция за създаване на информация, константно установяваща евентуално несъвпадение на части от низа.

Авторът е използвал , където *si* са стойностите на символите от низа, *i* се мени от 0 до *m*-1 (т. е., частичните суми на цифрите от началото до текущата с алтернативни знаци). Тогава, ако |*h*(*e*1)-*h*(*b*1-1)| ≠ |*h*(*e*2)-*h*(*b*2-1)|, частта от низа от символ с номер *b*1 до символ с номер *e*1 със сигурност не е една и съща с частта от *b*2 до *e*2. Разбира се, можем да сверяваме винаги с първите *c* символа, т.е., с ±*h*(*c*), където *c*=*e*1-*b*1. Очаква се колизиите да са редки, те подлежат на пряка проверка.

* Ако установим липса на период с дължина *c*, то, очевидно, никой от делителите на *c* няма да бъде период и те могат да се изключат от проверка.
* Напротив, ако установим период *c*, то търсеният най-малък период е делител именно на *c* и можем да изключим от разглеждане всички делители на *m*, които не делят *c*.
* И последно: ако има период *c*, на последваща проверка за някой от неговите делители *d* подлежи, разбира се, само част с дължина *c* от низа (например – началото му). Установеното за нея (наличие или липса на период), естествено, важи за целия низ.

Броят на всички делители на *m*, при ограниченията на задачата, не надминава 240. Алгоритъмът за тяхното намиране е със сложност . Целият процес се мажорира от O(*m*)=O(*N*2).

*Автор: Павлин Пеев*