Префиксни суми

February 19, 2020

1 Загрявка

Задача 1. *Намерете броя числа в интервала* $[l, r]$*, които се делят на 7.*
Ограничения: $1 \leq l \leq r \leq 10^{18}$.

Решение. Може да обходим и проверим всички числа в интервала, но ще е доста бавно. Може да го забързаме като намерим първото число кратно на 7 и после прескачаме през 7, но пак може да се наложи да проверим много числа.
Да разгледаме друг подход. Знаем, че всяко седмо число е кратно на седем. От там идва идеята да вземем броя числа в интервала и да го разделим на 7. Идеята е много добра, но не съвсем вярна - има интервали с по равен брой числа, но различен брой кратни на 7, например интервалите $[6, 13]$ и $[7, 14]$. Това, че всяко седмо число е кратно на 7, ще ни даде правилно решение, ако броя на числата в интервала е кратен на 7. Може да използваме това, като проверим последните няколко числа поотделно, и да намалим интервала така, че той да стане с кратен на 7 брой числа и да използваме формула за останалия интервал. Например в интервала $[20, 99]$ има 80 числа. За да получим кратен на 7 брой може да махнем последните 3. Така ще проверим за 99, 98 и 97, и ще използваме формула за интервала $[20, 96]$. 98 е кратно на 7, в $[20, 96]$ има $77/7 = 11$ кратни на 7, и общия отговор за интервала $[20, 99]$ ще е 12.
Друг подобен вариант е да намерим първото и последното число кратни на 7 и да използваме формула свързана с тях. Ако първото и последното число кратни на 7 са a и b, то броят на всички е $(b - a)/7 + 1$.
Последните два варианта решават задачата, но има доста случай за които трябва да се внимава. Също ако търсим кратни не на 7, а на нещо доста по-голямо ще трябва по-умно да намираме последното число в интервала кратно на даденото.
Целта на тази глава е да ви накара винаги когато видите нещо, което се търси в произволен интервал, да пробвате да го разбиете на два интервала, които започват от едно място, например 0 или 1. В тази задача в сила е следната важна връзка - броя числа кратни на 7 в $[l, r]$ е равен на броя числа кратни на 7 в $[1, r]$ минус броя числа кратни на 7 в $[1, l - 1]$. Иначе казано, ако преброим числата кратни на 7 от 1 до r, ще сме броили и тези

по-малки от l, за това трябва да ги извадим. Важно е да отбележим, че интервала, който вадим е до $l-1$, понеже самото l, не трябва го вадим.
Остава да видим как да сметнем броя числа кратни на 7 в интервала $[1, n]$.
Тук вече доста по-лесно може да съобразим, че отговора е точно $n/7$.
Следва имплементация на решението:

```
long long solve(long long l, long long r) {
    return r/7-(l-1)/7;
}
```

□

Задача 2. *Намерете броя числа в интервала $[l, r]$, които се делят на d.*
Ограничения: $1 \leq l \leq r \leq 10^{18}, 1 \leq d \leq 10^{18}$.

Решение. Идеята е същата, само сменяме 7 с d.

```
long long solve(long long l, long long r) {
    return r/d-(l-1)/d;
}
```

□

2 Префиксни суми

В тази глава всички редици с n елемента ще ги пазим в масиви с $n+1$ елемента. Ако масива е a, то елемента $a[0]$ ще бъде помощен и винаги ще има стойност нула. Елементите представляващи редицата ще са $a[1], ..., a[n]$.

Дефиниция 1 (Префиксна сума)**.** *Да разгледаме редицата $a_1, a_2, ..., a_n$. Всяка сума от вида $s_i = a_1 + ... + a_i$, която включва първите няколко последователни числа се нарича префиксна сума.*

Задача 3. *Дадена е редица с n числа - $a_1, a_2, ..., a_n$ и q заявки. За всяка заявка е дадено едно число k, и трябва да намерите сумата на подредицата $[1, k]$, т.е. $a[1] + a[2] + ... + a[k]$.*
Ограничения: $1 \leq n \leq 10^6, 1 \leq q \leq 10^6, 0 \leq a[i] \leq 10^6, 1 \leq k \leq n$.

Решение. Първо ще отбележим, че сумата на числата може да стане голяма и за това ще я пазим в променлива от тип *long long*.
Задачата има очевидно решение - за всяка заявка обхождаме всички числа $a[1], a[2], ..., a[k]$ и ги събираме. Това решение обаче не е за максимален резултат, понеже е бавно. За всяка заявка може да са необходими близо до n събирания, като умножим по q заявки, получаваме nq операции. Това е доста голямо число при дадените ограничения.
Понеже числата в заявките са ограничени до n, то ние може предварително да пресметнем всички отговори. В масива p ще пазим тези суми, т.е. $p[i] = a[1] + a[2] + ... + a[i]$, $p[i]$ е сумата на всички числа до i-тото. Ако имаме този

масив на всяка заявка k, отговорът ще е $p[k]$.
Сега трябва да запълним масива p. Отново имаме бавен вариант като той изисква за всяко i да съберем всички числа, участващи в $p[i]$.
Много лесно обаче може да забързаме нещата. Да разгледаме сумата $p[i] = a[1] + a[2] + ... + a[i-1] + a[i]$. Знаем, че $p[i-1] = a[1] + a[2] + ... + a[i-1]$ и може да заместим. Така стигаме до $p[i] = p[i-1] + a[i]$, което е идеално за нас. Иначе казано сумата на първите i числа е равна на сумата на първите $i-1$ числа плюс i-тото число. Така с едно обхождане на масива може да пресметнем всички суми от началото.
Тук е момента да отбележим, че при така намерената формула $p[i] = p[i-1] + a[i]$ имаме $p[1] = p[0] + a[1]$. Това $p[0]$, което използваме при смятането на първата префиксна сума е една от причините да имаме нулев елемент със стойност нула. Ако нямахме този първи елемент първата префиксна сума щеше да е $p[0] = p[-1] + a[0]]$ и щяхме да излизаме от масива.

```
#include<bits/stdc++.h>
using namespace std;

const int MAXN = 1000000;
int a[MAXN+1], p[MAXN+1];

int main() {
    int n, q;
    cin >> n >> q;
    for (int i = 1; i <= n; i++) {
        cin >> a[i];
    }
    for (int i = 1; i <= n; i++) {
        p[i] = p[i-1]+a[i];
    }
    for (int i = 0; i < q; i++) {
        int k;
        cin >> k;
        cout << p[k] << endl;
    }

    return 0;
}
```

□

Задача 4. *Дадена е редица с n числа - $a_1, a_2, ..., a_n$ и q заявки. За всяка заявка са дадени две числа l и r, и трябва да намерите сумата на подмасива $[l, r]$, т.е. $a[l] + a[l+1] + ... + a[r]$.*
Ограничения: $1 \leq n \leq 10^6, 1 \leq q \leq 10^6, 0 \leq a[i] \leq 10^6, 1 \leq l \leq r \leq 10^6$.

Решение. Сумата на числата отново трябва да е от тип *long long*.
По подобие на предната задача има бавно решение при което за всяка заявка

обхождаме всички числа $a[l], a[l+1], ..., a[r]$ и ги събираме.
Идеята за подобрение идва от това, че работим с интервали. Прилагаме първото правило при решаване на задачи с интервали $[a, b]$ - да проверим дали може да решим задачата като разлика на два интервала с общо начало. Да разгледаме редицата $a = \{8, 6, 3, 5, 7, 3, 9\}$ и да намерим сумата от 3ия до 6ия елемент, $a[3] + a[4] + a[5] + a[6]$. Да видим как може да използваме префиксните суми p, където $p[i] = a[1] + ... + a[i]$.

	0	1	2	3	4	5	6	7
a $\longrightarrow$	0	8	6	3	5	7	3	9

	0	1	2	3	4	5	6	7
p $\longrightarrow$	0	8	14	17	22	29	32	41

Лесно може да се усетим, че $a[3] + a[4] + a[5] + a[6] = (a[1] + a[2] + a[3] + a[4] + a[5] + a[6]) - (a[1] + a[2] + a[3]) = p[6] - p[3]$. Може и да го видим - сумата от числата в жълтите квадратчета е равна на разликата от числата в зеленото и червеното квадратче.
Може да обобщим, че $a[l] + ... + a[r] = p[r] - p[l-1]$, т.е. сумата на числата $a[l] + a[l+1] + ... + a[r]$ е равна на сумата на всички число от началото до r минус сумата на всички числа преди l.

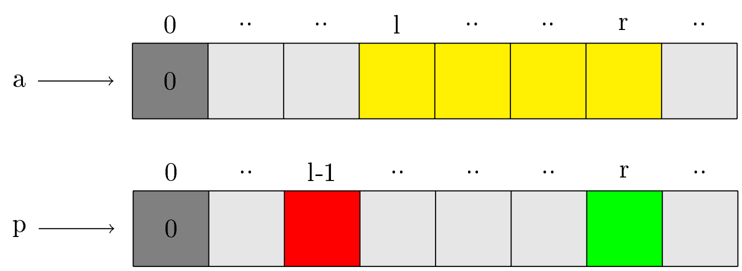
```
#include<bits/stdc++.h>
using namespace std;

const int MAXN = 1000000;
int a[MAXN+1], p[MAXN+1];

int main() {
    int n, q;
    cin >> n >> q;
    for (int i = 1; i <= n; i++) {
        cin >> a[i];
    }
    for (int i = 1; i <= n; i++) {
        p[i] = p[i-1]+a[i];
    }
    for (int i = 0; i < q; i++) {
        int l, r;
        cin >> l >> r;
        cout << p[r]-p[l-1] << endl;
    }
```

```
    return 0;
}
```

□

Задача 5. *Дадена е редица с n числа -* $a_1, a_2, ..., a_n$*. Намерете дали съществува последователност от числа със сума равна на нула.*
Ограничения: $1 \leq n \leq 10^6, 0 \leq a[i] \leq 10^6$.

Решение. Бавните решения включват проверка за всяка последователност. Например може да сметнем префиксните суми в началото и после за всяко възможно начало и край на последователност, чрез префиксните суми бързо да проверяваме дали сумата в текущия интервал е нула.
За по-бързо решение, да помислим как може да използваме префиксните суми $p[i] = a[1] + ... + a[i]$ в тази задача. Нека да си представим, че съществува последователност $a[l] + ... + a[r] = 0$.



Знаем, че $a[l]+...+a[r] = p[r]-p[l-1]$ и понеже разглеждаме последователност със сума нула, то $0 = p[r] - p[l-1]$, т.е. $p[l-1] = p[r]$. Това което получихме е, че ако има последователност с нулева сума, то има две равни префиксни суми. Лесно може да проверим, че и обратното е вярно. Т.е. нулева сума ще има тогава и само тогава, когато има поне две равни префиксни суми.
Възможно е да получим, че нашия специален елемент $p[0] = p[r]$. Това ще се случи ако имаме нулева сума от първия елемент до някъде, което означава, че трябва да имаме предвид и $p[0]$. Т.е. като проверяваме за равни префиксни суми, трябва да имаме гледаме и $p[0]$.
Сега остава да промерим дали има две равни префиксни суми. За целта може да ги сортираме и след това да проверим дали има две съседни равни числа.

```
#include <bits/stdc++.h>
using namespace std;

const int MAXN = 1000000;
int a[MAXN+1], p[MAXN+1];

int main() {
    int n;
```

```
    cin >> n;
    for (int i = 1; i <= n; i++) {
        cin >> a[i];
    }
    for (int i = 1; i <= n; i++) {
        p[i] = p[i-1]+a[i];
    }

    sort(p, p+n+1);

    bool hasZeroSum = false;
    for (int i = 1; i <= n; i++) {
        if (p[i] == p[i-1]) {
            hasZeroSum = true;
            break;
        }
    }

    if (hasZeroSum) {
        cout << "yes" << endl;
    } else {
        cout << "no" << endl;
    }

    return 0;
}
```

□

3 Суфиксни суми

Дефиниция 2 (Суфиксна сума). *Да разгледаме редицата* $a_1, a_2, ..., a_n$. *Всяка сума от вида* $s_i = a_i + a_{i+1} + ... + a_n$*, която включва последните няколко последователни числа се нарича суфиксна сума.*

Задача 6. *Дадена е редица с* n *числа -* $a_1, a_2, ..., a_n$ *и* q *заявки. За всяка заявка е дадено едно число* k*, и трябва да намерите сумата на подредицата* $[k, n]$*, т.е.* $a[k] + a[k+1] + ... + a[n]$.
Ограничения: $1 \leq n \leq 10^6, 1 \leq q \leq 10^6, 0 \leq a[i] \leq 10^6, 1 \leq k \leq n$.

Решение. Тази задача е много подобна на задачата за префиксните сума. Нека $s[i] = a[i] + ... + a[n]$. Връзката която може да съобразим е, че $s[i] = s[i+1] + a[i]$.
Понеже да сметнем $s[i]$ ни трябва $s[i+1]$, то трябва да попълваме масива s в обратен ред.
За първия суфикс имаме $s[n] = s[n+1] + a[i]$. Така както при префиксите

използвахме нулевия елемент за служебен, тук ще трябва да имаме един елемент след последния. За улеснение най-добре винаги да имаме два служебни - един в началото и едни в края.

```
#include <bits/stdc++.h>
using namespace std;

const int MAXN = 1000000;
int a[MAXN+2], s[MAXN+2];

int main() {
    int n, q;
    cin >> n >> q;
    for (int i = 1; i <= n; i++) {
        cin >> a[i];
    }
    for (int i = n; i >= 1; i--) {
        s[i] = s[i+1]+a[i];
    }
    for (int i = 0; i < q; i++) {
        int k;
        cin >> k;
        cout << s[k] << endl;
    }

    return 0;
}
```

□

Задача 7. *Дадена е редица с n числа - $a_1, a_2, ..., a_n$. За всяка позиция i намерете последната цифра на произведението на всички числа без i-тото, т.е за всяко i, намерете $(a[1] * ... * a[i-1] * a[i+1] * ... * a[n])\%10$. Ограничения:* $1 \leq n \leq 10^6, 0 \leq a[i] \leq 10^6$.

Решение. Първо ще отбележим, че при произведение резултата бързо ще надвиши *int* и *long long*. Понеже ни трябва последната цифра ще пазим винаги само нея. Дори след като прочетем първоначалните числа ще ги сменим с последната им цифри, понеже другите не оказват влияние.
За варианта да пазим произведението на всички числа и да делим на всяко ще ни трябват дълги числа и въпреки това пак ще е бавно.
Друго решение е за всяко число да умножим всички останали, което очевидно е бавно.
Понеже имаме интервали логично е да се запитаме дали може да използваме префиксни суми, като в тази задача те ще се префиксни произведения. Нека $p[i]$ е последната цифра от произведението на първите i числа - $p[i] = (a[1] * ... * a[i])\%10$. Като заместим в това, което търсим получаваме $(a[1] *$

$... * a[i-1] * a[i+1] * ... * a[n])\%10 = (p[i-1] * a[i+1] * ... * a[n])\%10$. Имаме произведението на всички числа преди $a[i]$, но сега ни трябва и произведението на всички числа след $a[i]$. Но всъщност това са суфиксните произведения. Така ако пресметнем и тях в масива s, бързо ще може да кажем, че последната цифра на произведението на всички числа без i-тото е $(p[i-1] * s[i+1])\%10$.
Понеже имаме произведение двата служебни елемента в началото и в края трябва да ги сложим да са равни на 1.

```
#include <bits/stdc++.h>
using namespace std;

const int MAXN = 1000000;
int a[MAXN+2], s[MAXN+2];

int main() {
    int n;
    cin >> n;
    for (int i = 1; i <= n; i++) {
        cin >> a[i];
        a[i] = a[i]%10;
    }
    p[0] = 1;
    for (int i = 1; i <= n; i++) {
        p[i] = (p[i-1]*a[i])%10;
    }
    s[n+1] = 1;
    for (int i = n; i >= 1; i--) {
        s[i] = (s[i+1]*a[i])%10;
    }
    for (int i = 1; i <= n; i++) {
        cout << (p[i-1]*s[i+1])%10 << endl;
    }

    return 0;
}
```

□

4 Задачи

Задача 8. *Дадена е редица с n числа - $a_1, a_2, ..., a_n$. Намерете номерата на всички елементи, за които сумата на числата преди тях е равна на сумата на числата след тях, т.е всички числа i, за които $a[1] + ... + a[i-1] = a[i+1] + ... + a[n]$. Може да считаме, че сумата на числата преди*

първия и след последния е равна на нула.
Ограничения: $1 \leq n \leq 10^6, 1 \leq q \leq 10^6, -10^6 \leq a[i] \leq 10^6, 1 \leq k \leq n.$

Задача 9. *Летен Турнир 2018, E група, E3.Редица*

Задача 10. *НОИ 3 2017, E група, E3. Редица от правоъгълници*

Задача 11. *НОИ 3 2015, D група, D4. Пропуснат множител*

Задача 12. *НОИ 2 2013, C група, C3. Думи*

Задача 13. *НОИ 3 2019, D група, D3. Поздрави*

Задача 14. *Втора контрола за младежкия национален 2018, C група, CK4. Баланс*