

(Зад 6) Даден е масив с естествени числа.
Кое е най-малкото естествено число,
което не се съвръща в масива?

* $A[1..n] \subset A[i] \in \mathbb{N} \forall 1 \leq i \leq n$.

Решение със сложност $\mathcal{O}(n^2)$

е всичко естествено число от

1 до n да проверим дали е някакде
в масива. Ако не се съвръши
значене $n+1$ е първото не съвр-
щано че.

Вариант с $\mathcal{O}(n \lg n)$ \rightarrow сортиране
и търсене $A[i] \neq A[i-1] > 1$.

Значи $A[i-1] + 1$ липсва.

Най-добър вариант е също
временно сложност $\mathcal{O}(n)$ е
следното:

Недоринирани зони на нули
наризи $B[1..n]$ от дясно.

Резултатът на използване това, че
неколкото отговори на B са
 $1, 2, 3, \dots, n$ или $n+1$, така че

и не иницирирае no-recurse
of n числа.

Е Counting sort опише
всички елементи в масива,
което $A[1..k] = n$.

След това нинейните
през всички възможни
най-рано $O \rightarrow$ инициализира
това съществуващо
число, n ,
иначе връщащо $n+1$.
Този алгоритъм назвава
съществително по мета $O(n)$.

Идеята е с конструиране
на място като се разглежда, че
числото в масива е естествено
число и можем да приемем
задачата за същото че задача
на инициализация, така че да разгледаме
за броят на.

Идеята е, че то инициализира
това е отрицателно, то

Значи сие средноряди члены
и в масиве $A[1..n]$.

Тръбва да внимаваме да
разделиме двинети подмножини
ст-ти. След това трябва и
да съберем непротивоположни членове
и да извадим член, който е отговорен
член $A[n+1]$.

Насъбодък:

$Qbs(A[1..n])$: array of notes : not
1. for $i \leftarrow 1$ to n do:
2. if $Qbs(A[i]) <= n$
3. $A[abs(A[i])] = -abs(A[abs(A[i])])$
4. for $i \leftarrow 1$ to n do:
5. if $A[i] > 0$ then
6. return i
7. return $n+1$.

Тога $T(n) \asymp O(n)$, а $Q(n) \asymp O(1)$.

Пример на сългободък
от държавата.

npusw: 8 100 1 5 3 9 6 4
 i=1 (1 2 3 4 5 6 7 8
 ↓
 8 (10) 1 5 3 9 6 -4
 i=2 (8 100 1 5 3 9 6 -4
 ↓
 i=3 (8 100 1 5 3 9 6 -4
 ↓
 i=4 (-8 100 1 5 3 9 6 -4
 ↓
 i=5 (-8 100 1 5 -3 9 6 -4
 ↓
 i=6 (-8 100 -1 5 -3 9 6 -4
 ↓
 i=7 (-8 100 -1 5 -3 -9 6 -4
 ↓
 i=8 (-8 100 -1 -5 -3 -9 6 -4
 ↓
 1 (2) 3 4 5 6 7 8
 ↑

(300,8) Начи с и е с т вен ческ ,
всю о/у 1 ии вх+степено. Ако се
комери нас ли повторяюще ческ .
(Помен да именование ени. В искубо.)

$A[1..n]$: $1 \leq A[i] \leq n$ за $1 \leq i \leq n$.

Вариант 1: Сортираме и
търсим $A[i-1] - A[i] = 0$ $O(n \lg n)$

Вариант 2: Знаем, че ческ е
са едък 1 ии. Зноти с
Counting Sort и допълнителен
начив без нострици
 $T(n) \approx O(n \lg n)$ $\Theta(n) \approx O(n)$.

Среща това линейни ческ
но допълнителни начив и,
ако има елемент > 1 , то
връщаме true.

Вариант 3: Като символи
записът ще бъде израз
от това, че са със нови ческ .
Съсист е същия!

Ако не носи, ческ е отриц.

Ці згадки є чистою речиною
всесвіту в науці. А їх
важливо не зупиняти згідно
з нападами експертів щодо
оригінальності та заслугування
нобільшів щодо їхніх.

Насправді:

Qbs(A[1..n]): array of ints; bool

1. for $i \leftarrow 1$ to n do
2. if ($A[\text{Qbs}(A[i])] > 0$)
3. $A[\text{Qbs}(A[i])] = -A[\text{Qbs}(A[i])]$
4. else
5. return true
6. return false

1 2 3 4 5 6 7
Input: 1 4 3 1 6 1 3
 $i=1$ { 1 4 3 1 6 1 3
 $i=2$ { -1 4 3 1 6 1 3
 $i=3$ { -1 4 3 -1 6 1 3
 $i=4$ { -1 4 -3 -1 6 1 3
 $\underline{-1}$
+ the

(300) Напиши масив от номинации и
щеми таас. Има ли сред тях две, които
разликата не е равна на?

$A[1..n] : \{A[i] \in \mathbb{N} \mid 1 \leq i \leq n\}$
 $i, j \in \{1, \dots, n\} : |A[i] - A[j]| \neq 1$

Вариант 1:

Да използваме практика, че
остатъците са от 0 до $n-1$ и
да използваме Counting
Sort.

Q1 > (A[1..n]: array of ints): bool

1. $B[0..n-1] : \text{array of } \{\text{bool}\}$.
2. // $B[i] = \text{true} \Leftrightarrow \text{без остатък}$
3. // OCTOBER
4. for $i \leftarrow 0 \text{ to } n-1$ do
5. $B[i] \leftarrow \text{false}$
6. for $i \leftarrow 1 \text{ to } n$ do
7. if $B[A[i] \bmod n] = \text{true}$
8. return true
9. $B[A[i] \bmod n] = \text{true}$
10. return false // Няма остатък

$$T(n) \leq n + n = 2n \leq n, \text{ и}$$

$$S(n) \leq n.$$

Итак, и с константой всё верно.

Вариант 2:

Рассмотрим вспомогательные операции, которые относятся к вычислению общего количества единиц в строке.

Итак, мы хотим найти количество единиц в строке, в которой есть хотя бы одна единица.

Ако $A[i] < 0 \Leftrightarrow$ остаток i не
делится на 2.

algo(A[1..n]: array of ints): bool

1. for $i \leftarrow 1$ to n do

2. if $A[\text{Qbs}(A[i]) \text{mod } n] < 0$

3. return true

4. $A[\text{Qbs}(A[i]) \text{mod } n] \leftarrow$

5. $-\text{Qbs}(A[\text{Qbs}(A[i]) \text{mod } n])$

6. return false

Всё верно $T(n) \leq n$ и $S(n) \leq 1$.

запомнил для i .

Задача от informatica.

Отделни файл с условието.

Изходът е, че трябва на всяка
страница, която ни подават
ново обучение, ние да разделим
подробностите и да използваме
неговата.

Узор 1:

Нашествияме възможните обучения
в съдържанието масив на нова
обучение и включваме в него и
такъв пътникът за експансионе е
 $O(1)$, то включвателен (от-д-от
Insertion sort е $O(n)$).

Узор на обработката:

Тук имаме вход и изход, това
че нека с $in(\star)$ е от-д-от за
вход започвайки в организацията
вход, а $out(\star)$ го изпраща
на изхода.

Q1>()

1. in(O(n))

2. A[1...n]

3. for i ← 1 to n do

4. in(A[i])

5. insert(A[1...i], i)

6. sum ← A[i/2] + A[i/2] + i(mod 2 ≠ 1)

7. out(└sum/2┘ + (sum(mod 2) ≠ 1) * 0, 5)

insert(A[1...m]: list of nums, idx : num)

1. while idx > 0 ∧ A[idx - 1] > A[idx] do

2. swap(A[idx - 1], A[idx])

3. idx ← idx - 1

Най-хорош сложност: $O(n^2)$ време и $O(n)$ памет.

Можем ли създавате хъм?

Идея е създавати:

User 1:

където разделят гъба heaps

т.е. приоритетни списъци.
Списъкът е MinHeap, а
записът MaxHeap и така
недействителни списъци.

Нека при новите списъци
елементът си е ~~във~~ записан във
списъците и #елементът ще
получи като нека. е 1-ият.
 като ~~във~~ записан елемент.

За какъдълък ще ни хепс?

В MinHeap некоето ще започне
със ~~нека~~ като -значение елемент.
но ~~нека~~, а в MaxHeap ще
се ~~нека~~-значението елемент ще е ~~нека~~.
както ~~нека~~ трябва да е #елемент.

Нека. с 1-ия ще се различават.

- Ако ~~нека~~ ще си ~~във~~ записан
елемент ~~нека~~, то нека в
MinHeap ще са ~~нека~~. Така е написано
- Ако ~~нека~~ ще ~~във~~ записан
елемент ~~нека~~ ще е същото.

За момент кога си познакум
за корупция и воровство
генофондите им да се раздават
с 1-ко, но и така беше кога
и нон фамилии.

Искаме очаквахме да е възможно
и след това да видим как
се е променила ситуацията.

Гледахме Minkeap.r.tg и там е
написано от него, че е възможност
да се извадят, че се възстанови
Minkeap, че се възстанови.

Т.е. сравнихме със кой-комуници
от кой-големите. Ако е по-голям
от него отива при големите,
които го приличат при големите.
Сега искаме и да се извадят
и воровствата и #елементи
онаквите с 1-ко както да
се раздават.

Ако е така, то гледахме въ
този онакъв и да се извадят

елементи. Този метод е MinHeap.

Този метод наричава се `MinHeap.top`.

В `MinHeap` имаме `MinHeap.top` и `MinHeap.push`.

Ако искаме да извадим

съдържанието на `MinHeap`,

Създаваме нов `MinHeap` за

неговите, то ги изваждаме от него.

- Ако `MinHeap` има нобещи
елементи връщаме `MinHeap.top`

- Ако `MaxHeap` има нобещи
елементи връщаме `MaxHeap.top`

- Инака започваме с реден #
елемент и връщаме

(`MaxHeap.top + MinHeap.top`) / 2.

Изтегляне `Heap.top` е $O(1)$

Сложността по време, определяне,

тъй като елемент съществува

сложността по време $O(\log n)$.

Задади ищаме сложността по

време $O(n \cdot \log n)$, която не е

така е $O(n)$. Съществува константа,
която ик懈.