**divisors0 и divisors1**

Най-лесното решение е за всеки префикс да питаме за всички негови делители. От страна на имплементация обаче е по-лесно за всеки потенциален период да питаме за всички префикси с дължина кратна на периода. Това решение изкарва само 6 точки, но с една лесна оптимизация стига до 13 точки: разглеждаме възможните периоди във възходящ ред и не питаме за префикси с вече открити периоди.

**divisors2**

Нека дефинираме „частичен“ период по същия начин като период, но без изискването за делимост. Лесно се вижда (от дефиницията), че ако позиция има частичен период , то и всички позиции преди също го имат. Следва, че ако позиция няма частичен период , то и всички позиции след също го нямат. Също така забележете, че ако позиция има период , то тя има и частичен период .

От горните наблюдения можем да видим, че ако позиция няма период , то и позиция също няма да има период . Следва, че можем да разглеждаме позициите във възходящ ред и, когато стигнем до , което няма период , можем да прекъснем търсенето. Това решение ни носи 47 точки.

**divisors3**

Нужното наблюдение тук е, че ако позиция има период , то не е минималният период на никоя следваща позиция, т.е. можем да не правим търсене за период . Това е защото, ако следваща позиция има период , тя със сигурност ще има и период . Това решение получава 69 точки.

**divisors4**

Нека позиция има минимален период . Следва, че всички позиции до имат частичен период . Сега, нека разгледаме позиция между и , за която твърдим, че има минимален период . Тя обаче има и частичен период . Лесно се вижда, че от това следва следното: има период (може да се опитате да си разпишете контрапример). Ако не дели , то , която е противоречие, защото приехме, че е минималният период на . Следва, че трябва да дели . От това обаче следва, че позицията трябва също да има период (тъй като и има частичен период , също има частичен период ). Но това води до противоречие по съображенията направени в предното решение. Заключваме, че никоя позиция между и няма минимален нетривиален период .

Сега нека да приемем, че позиция има период , такъв че . Разглеждаме следните варианти. Първо, ако , стигаме до противоречие от горния тип на позиция , защото . Второ, ако , стигаме до противоречие на позиция (с ролите на и разменени), защото . Така заключваме, че винаги се стига до противоречие, т.е. никоя позиция не може да има период между и . (Забележете, че случаят бива изключен още от съображенията направени в предното решение.)

Стигаме до следното решение: правим това, което прави и divisors3, но когато прекъснем търсенето за период с финална открита позиция , следващият период, който разглеждаме, не е , ами е . Това решение изкарва 84 точки.

**Анализ на divisors4**

Нека сега анализираме броя заявки, които това решение извършва. В началото си „заделяме“ заявки, по една за всеки възможен период. Когато разглеждаме период , който е минимален на позиция , използваме общо заявки, но пък елиминираме всички периоди до , както и (поради съображенията направени в предното решение). При , единственият ново-елиминиран период е , но това е в „бюджета“ ни, защото за него има заделена една заявка. При , общата бройка ново-елиминирани пероди е , т.е. отново се придържаме към бюджета си. Всичко това ОБАЧЕ се случва само докато текущата позиция е преди , защото иначе няма периоди за елиминиране. Ако няма нетривиални периоди след позиция , това не е проблем и се побираме в заделените си заявки.

Сега нека разгледаме случая да има нетривиален период след позиция . Това значи, че не може да има други нетривиални периоди след него. В най-лошия случай са ни трябвали заявки за целия масив. Първите се използват преди достигане на периода (най-лошият случай е всички периоди преди да са единици), а следващите – след това (най лошият-случай е да питаме за всяка кратна на позиция след ).

Лесно се вижда, че тази функция на е изпъкнала, т.е. максимумът ѝ в някакъв интервал е равен на стойността ѝ в един от двата края на интервала. Ясно е, че . Следва, че това решение прави не повече от заявки, ако приемем, че . За жалост при , то може да достигне заявки. Т.е. лошият му случай е, когато над половината от масива има период равен на едно.

**divisors5**

Горният анализ ни показва, че предното решение се „чупи“ в само един много конкретен клас от случаи. Най-лесното решение на този проблем веднага трябва да ни е очевидно: като първа заявка ще проверим дали сме в лош случай, т.е. ще проверим дали позиция има период равен на едно. Ако не, то не сме в лош случай и можем да процедираме както преди. Ако да, то знаем, че всички позиции до имат единичен период и остава линейно да проверим за всички следващи (общо заявки за тази дясна част). И в двата случая правим не повече от заявки. Това е пълното решение.

**Динамично програмиране**

Едно частично решение, предложено от Радослав Димитров, се базира на идея напомняща на динамично програмиране. Ако позиция има минимален период , то и позиция трябва да има минимален период . Така можем да инициализираме периодите на всички позиции да са равни на самите им индекси. След това, когато достигнем до позиция (по индукция) вече знаем минималният ѝ период . Остава да проверим дали това е период и на позиция и, ако да, ъпдейтваме периода ѝ. Забележете, че преди това ъпдейтване (преди проверката дори) текущият минимален период на винаги ще е равен на , т.е. никога няма да ъпдейтваме дадена позиция над веднъж. Това позволява ъпдейта да е просто присвояване на стойността на , а не ъпдейтване като минимума на и предишната стойност. Това е следствие на наблюденията ни за divisors4, но тук не се възползваме от тях и затова това решение не е толкова добро. То изкарва 69 точки, а с оптимизация подобна на тази за divisors5 – 83 точки.

**Долна граница**

Вече разгледахме как да решим задачата, но дали това наистина е най-доброто възможно решение? Можем ли да покажем, че не е възможно да съществува по-добро решение? Ами, почти! Можем лесно да покажем долна граница от заявки. Да разгледаме следния случай: има не повече от една позиция с нетривиален период. Очевидно тази позиция трябва да е четна (периодът се повтаря точно два пъти), но не можем да заключим нищо повече за нея. Следва, че не можем да я намерим по по-ефикасен начин от това да питаме за всички четни позиции дали имат период . Това отнема заявки за тази опростена версия на пълната задача, т.е. и пълната задача не може да се реши в под заявки. Разумно е да допуснем, че по-сложната пълна задача изисква поне още една допълнителна заявка (общо ), в който случай нашето решение би било оптимално, но нямаме доказателство за това. Във всички случаи то е много близо до оптимално.

Автор: Емил Инджев

Начална идея: Радослав Димитров