 ***ОТКРИТО ПЪРВЕНСТВО НА СОФИЯ ПО ИНФОРМАТИКА***

***15 ноември 2021 г.***

***Група C, 7-8 клас***

**Анализ на задача Кули**

Накратко трябва да намерим броя на редиците , от естествени числа, за които сумата на елементите е равна на S и е изпълнена една от следните зависимости:



**Първа подзадача.** Чрез брут форс можем да генерираме всички възможни редици със сума на елементите S и да проверим кои от тях изпълняват една от двете зависимости. Сложност – **O( SS ).**

**Втора подзадача.** Можем да подобрим пълното изчерпване от предишната подзадача, като генерираме само валидните редици. Нека е броят на редиците със сума , за които първия елемент е строго по-голям от за или строго по-малък от за . Знаейки, какви условия трябва да са спазени за първия елемент, лесно получаваме следната зависимост:

Отговора на задачата ни ще е – броят на редиците от вида > < > ... плюс броят на редиците от вида < > < ... , за които първия елемент е по-малък от S, защото иначе ще броим редицата {S} два пъти. Сложност – O( отговора ), което за е не повече от някъде 50млн стъпки.

**Трета подзадача.** За тази подзадача няма предвидени конкретни решения, тя е за тези, които не са написали решението си, така че да работи с оптимална скорост или за тези, подобрили решението на предишната подзадача с някакви оптимизации.

**Четвърта подзадача.** Веднъж щом видим формулата от втора подзадача, няма как да не се сетим и за динамичното програмиране. Параметрите са достатъчно малки, за да попълваме намерените отговори в един масив .

Сложност – **O( S3 ).**

**Пета подзадача.** Следващата оптимизация след мемоизацията ще е съкращаване на рекурентната формула. Да се загледаме във формулите на два доста близки стейта:

Всички събираеми без едно съвпадат. Можем да се възползваме от този факт и да запишем формулата по следния начин:

Разсъждения за са същите:

Така си спестяваме линейните цикли за всеки стейт. Сложност – **O( S2 ).**

**Шеста подзадача.** Какво още можем да подобрим? Рекурсията сама по себе в някои случаи може да бави решението. За тази подзадача е предвидена итеративната имплементация на динамичното. Така казано може и да звучи просто, но има някои подробности, които си струва да се споменат. Основният проблем при итеративните решения е, че трябва да преценим в какъв ред да запълваме стойностите в масива. Тук за дадено трябва да сме изчислили всички с по-малък . А ? При него трябва да имаме в предвид стойността на . Ако тя е 0, то трябва да ги изчисляваме в намаляващ ред на и в противен случай – в нарастващ ред на .

|  |
| --- |
| for(int last=1;last<=s;last++){  dp[0][last][0]=dp[0][last][1]=1;  }  for(int sum=1;sum<=s;sum++){  for(int last=sum-1;last>=0;last--){  dp[sum][last][0]=dp[sum-(last+1)][last+1][1] + dp[sum][last+1][0];  dp[sum][last][0]%=mod;  }  for(int last=2;last<=s;last++){  dp[sum][last][1]=dp[sum][last-1][1];  if(sum-(last-1)>=0)dp[sum][last][1]+=dp[sum-(last-1)][last-1][0];  dp[sum][last][1]%=mod;  }  } |

*Автор: Александър Гатев*