## Задача СКОКОВЕ

### Пояснение към решението

**Бавно решение (рекурсия)**

Прочитаме в масива `a[]` дадената редица. След това използваме рекурсивната функция `int run(int i)`, която пресмята минималния брой скоковете, за да достигнем индекс `n-1`, тръгвайки от индекс `i`. Намирайки се в индекс `i`, може да отидем в някой от следващите индекси от `i+1` до `i+a[i]`. Рекурсивно пресмятане `v=run(j)` за тези следващи индекси и ако може да се достигне такъв (т.е. `v != MAX`), подобряваме стойността, която ще пресметне извикването `run(i)`. В главната функция на програмата извикваме `run(0)`.

**По-бързо решение (рекурсия с мемоазация)**

Използваме рекурсията от бавното решение, но добавяме запомняне в масива `memo[i]` на вече пресметнатите стойности от извикванията на `run(i)`. В началото масивът `memo` е зареден със стойности `-1`.

**По-бързо решение (грийди)**

Реализира се чрез функцията `minJumps`. В тази функция тръгваме от индекс 0 и скачаме максимално:

```
maxReach = max(maxReach, i+a[i]);}
```

Ако при `i=0` пресметнатото е такова, че `maxReach >= n-1`, решението на задачата е равно на 1. В противен случай записваме `currReach=maxReach`.

Продължаваме в цикъла с индексната променлива `i` за `i>0`. Във всяка стъпка имаме пресметнати стойностите на `jump`, `currRach` и `maxReach`. Ако с един скок може да достигнем края на редицата, прекратяваме цикъла и отпечатваме `jump+1`. В противен случай продължаваме пресмятането само, ако `i == currReach`. Тогава или не може да се движим повече (`if (i == maxReach) return -1;`) или ако не е достигнат края, увеличавам брояча `jump` и актуализираме `currReach = maxReach`. Накрая програмата отпечатва това, което е пресметнала функцията `minJumps`.

**Бързо решение (грийди с еднопасово преминаване през входа)**

Забелязваме, че в решението с грийди използваме последователно и еднократно стойностите от масива `a[]`. Това означава, че може да нямаме масив в програмата, а да четем от входа тези последователни стойности. Така липсата на голям масив значително намалява паметта.

```
for (int i = 0; i < n; i++)
  {
    int a;
    if(i==0) a=a0; else cin >> a;
    maxReach = max(maxReach, i+a);
    if (maxReach >= n-1) return jump + 1;
    if (i == currReach)
     {
       if (i == maxReach) return -1;
       else { jump++; currReach = maxReach;}
     }
   }
```

*Емил Келеведжиев*