**Анализ на решението на задача  
C#. ЯЖ, СПИ, … - ТИЯ ТРИ РАБОТИ**

Нека означим с *Ai* броя начини, по които бебето може да порасте до *i* грама. Очевидно, че всеки от тези начини може да доведе и до тегло *i*+*p*, ако бъде допълнен с едно кърмене. Или казано с други думи от тегло *i* може да се премине към тегло *i*+*p*. Обаче от тегло *i* може да се премине и към тегло *i*-*q*+*p*, ако бебето първо цапа пеленките (олеква с *q*) и после го кърмят. Да отбележим, че ограничението за максимум едно цапане на пеленки между две кърмения предполага след всяко цапане непременно да има кърмене. В тази връзка последователните цапане и кърмене може да се разглеждат като едно комбинирано действие за увеличаване на теглото. И така достигаме до следната рекурентна зависимост:

(1)

която може да реализираме рекурсивно (вероятно с мемоизация) като съобразим, че дъното на рекурсията е

Горната рекурентна зависимост може да се реализира и итеративно, като се проследи обратния ход на рекурсията, започвайки от дъното, което е същността на техниката динамично оптимиране. Понеже разглеждаме начините за достигане до дадено тегло в нарастващ ред на теглата, първо ще достигнем до решението *Aj*-*p* (теглото *j-p* е по-малкото). Това е първото събираемо в (1) и можем да го запазим като начална стойност за *Aj*. Когато достигнем до теглото *j-p+q* ще добавим решението за него   
*Aj*-*p+q* към *Aj*, с което ще намерим исканото решение според (1). Накрая трябва да съобразим, че тегла близки до *N*, при увеличаването им било с *p*, било с *p-q*, могат да доведат до тегла по-големи от *N*. Решенията *A* за такива тегла трябва също да се добавят към решението *AN*.

Сорс-кодът на програмата:

#include <cstdio>

**unsigned** A[8000]={1}, m,n,p,q,i,k,s,MOD=1<<30;

**int** main(){

scanf(" %u %u %u %u", &m, &n, &p, &q);

**for** ((n-m<p?n=0:n-=m+p), k=p-q; i<n; i++)

(i+p<n?A[i+p]=A[i]:s=(s+A[i])%MOD),  
 (i+k<n?A[i+k]=(A[i+k]+A[i])%MOD:s=(s+A[i])%MOD);

**return** printf("%u\n",s?s:1),0;

}

*Автор: Евгений Василев*