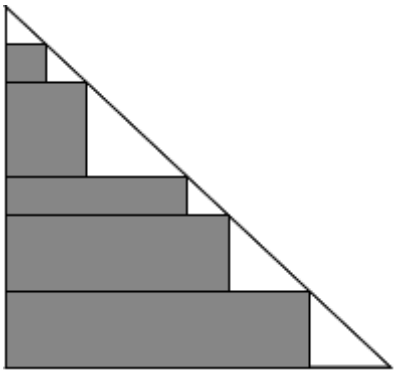
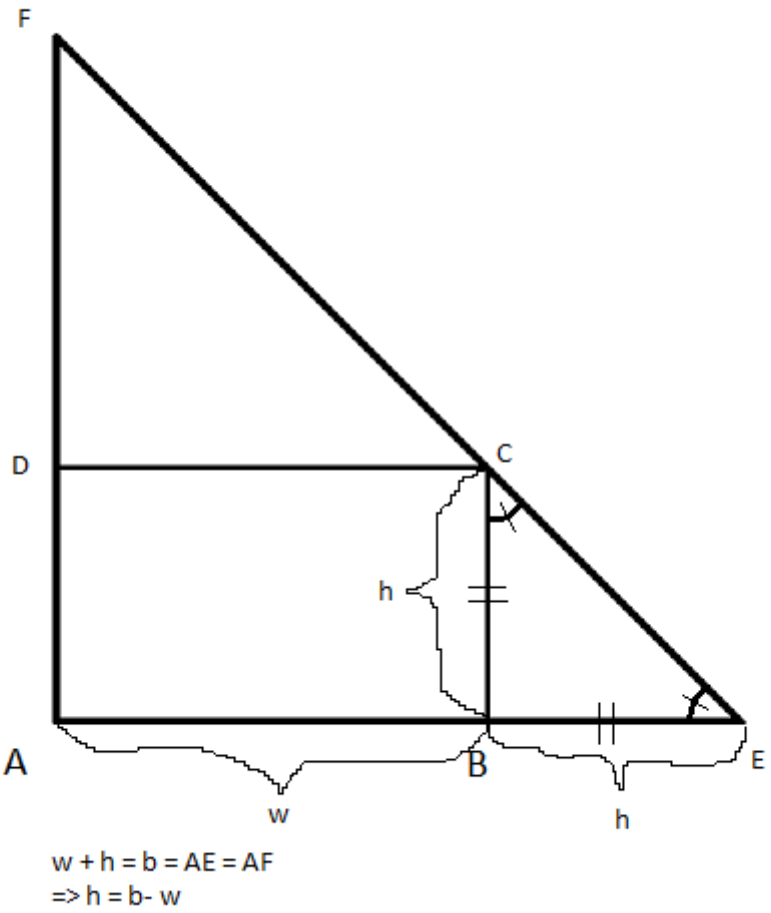
# Правоъгълници в триъгълник

В равнобедрен правоъгълен триъгълник са вписани (без припокриване) N правоъгълника, както е показано на рисунката.



Правоъгълниците имат целочислени координати на върховете си и са със страни, съответно успоредни на катетите на триъгълника, а върховете им лежат върху страните на триъгълника. Напишете програма, която въвежда N и дължината B на катета на триъгълника (цяло положително число, по-малко от 2000), и извежда лицето на най-голямата площ, която може да се покрие с правоъгълниците.

Жокер: Още една класическа задача от сферата на динамичното програмиране. Ще ползуваме двумерна динамична таблица DP[1..N][1..B], като първото измерение ще ни е броя на правоъгълниците 1..N, второто ще ни е дължината на двата катета на равнобедрения правоъгълен триъгълник. Ще инициализираме всички DP [i][j] с -1 означаващо че все още не сме изчислили резултата за нито една двойка (брой на вписани правоъгълници, дължина на катета). Също така знаем че когато в даден триъгълник искаме да впишем нула правоъгълника, то тяхната площ ще е 0. Следователно ще инициализираме DP[0][1..B] с 0 - това са граничните стойности на нашата таблица, които се явяват едновременно и условие за край на рекурсията. Най-лесно ще си представим попълването на DP използувайки рекурсия (unsigned long long calc_recursive(int N, int B)):




Намираме се на дадено ниво от рекурсията и трябва да пресметне колко е максималната сума от лицата на N вписани (един над друг) правоъгълника в равнобедрения праовъгълен триъгълник със страна B. Ако резултата е вече пресметнат (DP[N][B] != -1) го връщаме (return DP[N][B]) и излизаме от функцията. В противен случай генерираме всички възможни вписани правоъгълници със ширина w = 1..B-1 и височина B-w (виж горната фигура, защо h = B-w) за дадения равнобедрен праовъгълен триъгълник със страна B. Всеки вписан правоъгълник отрязва два триъгълника - BCE и CDF. С BCE не можем да направим нищо, но в CDF по условие трябва да впишем останалите N-1 правоъгълника. Но колко е тяхната сумарна площ left_area? И тук идва най-тънкия момент в задачата: left_area = calc_recursive(N-1, w), тъй като CD = AB = w. Вече за всеки вписан правоъгълник можем да пресметнем площта, която заемат всичките N вписани правоъгълника при условие, че той е най-долния от тях. Тя е FullArea[w] = (площта на текущия вписан правоъгълник с широчина w и височина B-w) + (площта на горните N-1 правоъгълника вписани в CDF) = w * (B - w) + calc_recursive(N-1, w). Имайки всички площи FullArea[w] за w = 1..B-1, вече сигурно

се досещате че най-голямата от тях ( max(FullArea[w] | w = 1..B-1) ) е търсеният резултата за calc_recursive(N, B). Записваме резултата в динамичната таблица (за да избегнем бавното му повторно изчисляване) ( DP[N][B] = max(FullArea[w] | w = 1..B-1) ) и го връщаме (return DP[N][B];), като резултат на текущото извикване на calc_recursive(N, B).

Целият този подход с динамично програмиране работи, защото страните на триъгълника и на вписаните правоъгълници са цели числа и могат да бъдат използувани при индексиране в таблицата DP.

Възможна оптимизация по време (едва ли ще ви трябва): Не е нужно да смятата FullArea[w], когато w<h, тоест когато w<B-w. Защо?

**Input Format**

Програмата трябва да прочете от стандартния вход броя на тестовите примери T (не повече от 10), след което – данните за всеки тестов пример от отделен ред, съдържащ N и B, разделени с интервал.

**Constraints**

0 < T <= 10
0 < N < 200
0 < B < 2000

**Output Format**

На стандарния изход трябва да се изведат търсените лица, всяко на отделен ред, съответно на входните данни.

**Sample Input 0**

```
3
1 1
1 2
2 10
```

**Sample Output 0**

```
0
1
33
```