Сума

Даден е списък с N цели числа A1, A2, ..., AN. Може да си изберете някакво цяло число K и към всяко от числата Ai да прибавите i * K. Така A1 става A1 + 1*K, A2 става A2 + 2*K, ..., AN става AN + N*K. Въпросът е: колко най-малко трябва да е K, за да съществува отрязък от списъка (тоест един или повече негови последователни елементи), чиято сума е поне M.

Нека, например, N = 5, M = 21, и първоначалните числа в списъка са (-5, 4, -13, -3, -7). Така, избирайки K = 3, числата биха станали (-2, 10, -4, 9, 8), като сумата 10 – 4 + 9 + 8 е по-голяма или равна на 21 (числото M). Може да се убедите, че това е и най-малкото K, при което това е възможно. Наистина, при K = 2 числата в списъка стават (-3, 8, -7, 5, 3), като никоя част от него не е по-голяма или равна на 21.

ЖОКЕР: За да успеете да решите тази задача задължително трябва да сте решили "Максимална сума от поредни елементи", тъй като тя се явява подзадача в тази. Ще използуваме стратегията разделяй и владей и ще разбием задачата на две подзадачи:
Подзадача 1) Да пресметнем при фиксирано K (FixedK) дали може да се получи подредица със сума поне M. Извикваме решението на "Максимална сума от поредни елементи", но с масива от числа [A1 + 1*K, A2 + 2*K, ..., N*K] вместо с [A1, A2, ..., AN]. От него получваме MaxSum и ако тя е >= M функцията за подзадача 1 връщаме true (възможно е при K = FixedK да получим сума поне M на някоя от всички възможни подредици), в противен случай (MaxSum < M) връщаме false.
Подзадача 2) Да намерим тази стойност на K, при която подзадача1(K) = true, а подзадача1(K-1) = false. Това ще ни даде най-малката стойност на K, при която може да се получи сума на подредица поне M.
Наивна стратегия: Да пробваме всички K= MIN_K.. MAX_K (намерете сами колко е MIN_K и MAX_K: MIN_K е тази стойност при която със сигурност всички елементи на редицата Ai + i * K ще са неположителни, а MAX_K ще е тази при която със сигурност всички елементи на редицата Ai + i * K ще са по-големи или равни на M, което от своя страна е максимум 1,000,000,000). Но в този случай бихме извършили максимално N * (MAX_K - MIN_K), което според мен е 10,000 * (1,002,000,000) ~= 10^13. Това число е много по-голямо от 10^9 и няма да влезете по време.
Оптимизирана стратегия: ДВОИЧНО ТЪРСЕНЕ с предикатна (bool) функция (около една или две от около 12 задачи от всяка олимпиада по програмиране се решават с този метод). Който не е запознат с базовия алгоритъма за ДВОИЧНО ТЪРСЕНЕ в

сортиран масив, задължително да се консултира на следните адреси:
Програмиране = ++Алгоритми (Programming = ++Algorithms) - страница 246
http://www.informatika.bg/lectures/binary-search
https://www.geeksforgeeks.org/binary-search/
https://en.wikipedia.org/wiki/Binary_search_algorithm
Дефинираме целочислен интервал [LEFT_K, RIGHT_K] на възможните отговори за K. В началото той ще е [LEFT_K=MIN_K, RIGHT_K=MAX_K]. На всяка стъпка ще стесняваме интервала на половина, докато LEFT_K < RIGHT_K и когато те станат равни това ще е отговорът на задачата. Как да стесним интервала? Ще използуваме подзадача 1. Вземаме текущото средно MIDDLE_K = (LEFT_K + RIGHT_K - 1) / 2. Ако подзадача1(MIDDLE_K) == true то ние не се интересуваме от стойностите на K > MIDDLE_K, тъй като те няма да са минималната възможна стойност за K и стесняваме интервала на [LEFT_K, MIDDLE_K], като правим RIGHT_K = MIDDLE_K. Ако подзадача1(MIDDLE_K) == false то ние не се интересуваме от стойностите на K <= MIDDLE_K, тъй като те биха имали суми за всяка от подредиците по-малки или равни на сумата на същата подредица за K=MIDDLE_K и също като MIDDLE_K няма да могат да постигнат сума M. В този случай стесняваме интервала на [MIDDLE_K+1, RIGHT_K], като правим LEFT_K = MIDDLE_K+1. Завъртаме цикъла, докато RIGHT_K > LEFT_K. Когато станат равни, това е отговорът на задачата ни.
Сложността на алгоритъма е O(N*log2(MAX_K - MIN_K)), което се равнява на около 10,000 * log2(1,002,000,000) ~= 10,000 * 30 = 300,000 за един тест. 300,000 * 25 теста = 7,500,000 е много по-малко от 10^9.
Ако не се сещате колко са MIN_K и MAX_K, то даже и MIN_K = (MIN_NEGATIVE_LONG_LONG + 1,000,000)/10000 и MAX_K = (MAX_POSITIVE_LONG_LONG - 1,000,000)/10000 ще ви свършат работа. Двоичното търсене лесно ще ги сведе до максимум 64 (вместо 30) стъпки за откриване на K. Прибавям/вадя 1,000,000 и деля на 10000, за да не стане препълване, когато AN става AN + N*K.

Input Format

На първия ред на стандартния вход ще бъде зададено цялото число T – броя тестове, които вашата програма трябва да обработи. Всеки тест е зададен на два реда. На първия от тях са дадени целите числа N и M – съответно броя числа в списъка и минималната търсена сума. На следващия ред са зададени N на брой цели числа A1, A2, ..., AN – първоначалните числа в списъка.

Constraints

$1 \le T \le 25$
$1 \le N \le 10,000$
$1 \le M \le 1,000,000,000$
$-1,000,000 \le Ai \le 1,000,000$

Output Format

За всеки тест на отделен ред изведете минималната стойност на K, която удовлетворява изискването.

Sample Input 0

```
2
5 21
-5 4 -13 -3 -7
10 42
5 17 -4 13 0 0 21 17 11 19
```

Sample Output 0

```
3
-1
```