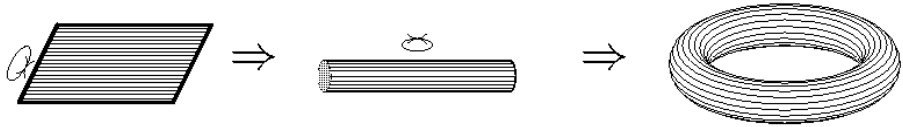
Задача A «Тор» (250 балів)

Як відомо, *тор* — це поверхня бублика, яку можна отримати таким чином: узяти прямокутник розміром n клітинок по вертикалі на m клітинок по горизонталі, склеїти верхню сторону з нижньою (отримається циліндр), потім закрутити циліндр у бублик і склеїти ліву сторону з правою.



Назвемо дві клітинки *сусідніми*, якщо вони мають спільну сторону. Нехай за одну секунду можна перейти з клітинки до будь-якої сусідньої з нею. За який мінімальний час можна потрапити з клітинки (r_1; c_1) у клітинку (r_2; c_2)? Перше число у позначенні клітинки — номер рядка початкового прямокутника, друге — номер стовпчика.

Вхідні дані Програма повинна прочитати зі стандартного входу (клавіатури) шість натуральних чисел, у порядку n, m, r_1, c_1, r_2, c_2.

Результат Програма має вивести на стандартний вихід (екран) єдине ціле число — мінімальний час.

Приклади введення і виведення:

Стандартний вхід (клавіатура)	Стандартний вихід (екран)
10 10 5 5 1 1	8
10 10 9 9 1 1	4

Обмеження $2 \le n, m \le 1\,000\,000\,000$, $1 \le r_1, r_2 \le n$, $1 \le c_1, c_2 \le m$.

Обмеження на час роботи програми: 1 секунда

У програмах категорично заборонено:

- Писати будь-які повідомлення, як-то наприклад “Vvedit’ N”. Програма повинна ***відразу***, без будь-яких запрошень, читати вказані в умові задачі вхідні дані. Знайшовши результат, программа повинна вивести ***тільки*** його. Наприклад, повинно бути ні в якому разі ***не*** `writeln('Kilkist sposobiv ', result)`, а тільки `writeln(result)`.
- Робити будь-яку затримку (байдуже, чи через зайві, непотрібні для читання реальних вхідних даних `read`-и або `readln`-и, чи через `repeat until keypressed`, чи ще якось — заборонені всі способи).
- Писати `uses crt`

Недотримання даних вимог автоматично призводить до неможливості перевірити програму автоматичними засобами, а отже — до оцінки в 0 балів.

Задача B «Паркет–1» (250 балів)

Щоб зобразити за допомогою паркету Супер-Креативний Візерунок, треба N_1 дощечок розмірами 1×1, N_2 дощечок розмірами 2×1, N_3 розмірами 3×1, N_4 розмірами 4×1 та N_5 дощечок розмірами 5×1. Купити можна лише дощечки розмірами 5×1. Дощечки можна різати, але не можна склеювати. Наприклад, коли потрібні п'ять дощечок 2×1, їх не можна зробити з двох дощечок 5×1, але можна з трьох. Для цього дві з них розріжемо на три частини 2×1, 2×1 та 1×1 кожну, а третю — на дві частини 2×1 та 3×1. Отримаємо потрібні п'ять дощечок 2×1, а дві дощечки 1×1 та одна 3×1 підуть у відходи.

Завдання Напишіть програму, яка, прочитавши кількості дощечок N_1, N_2, N_3, N_4 та N_5, знайде, яку мінімальну кількість дощечок 5×1 необхідно купити.

Вхідні дані слід прочитати зі стандартного входу (клавіатури). Це будуть п'ять чисел N_1, N_2, N_3, N_4 та N_5 (саме в такому порядку), розділені пропусками (пробілами).

Результат Єдине число (скільки дощечок треба купити) виведіть на стандартний вихід (екран).

Приклади введення і виведення:

Стандартний вхід (клавіатура)	Стандартний вихід (екран)
0 5 0 0 0	3
1 1 1 1 1	3

Обмеження — усі кількості невід'ємні; 90 балів припадатиме на тести, в яких сумарна кількість $N_1+N_2+N_3+N_4+N_5$ перебуває в межах від 0 до 20, ще 80 балів — від 100 до 10 000, решта 80 балів — від 500 000 000 до 2 000 000 000.

Здати потрібно *одну* програму, а не для кожного випадку окремо; різні обмеження вводяться виключно для того, щоб дати приблизне уявлення, скільки балів можна отримати, розв'язавши задачу не повністю.

Обмеження на час роботи програми: 1 секунда

У програмах категорично заборонено:

- Писати будь-які повідомлення, як-то наприклад "Vvedit' N". Програма повинна ***відразу***, без будь-яких запрошень, читати вказані в умові задачі вхідні дані. Знайшовши результат, программа повинна вивести ***тільки*** його. Наприклад, повинно бути ні в якому разі ***не*** `writeln('Kilkist sposobiv ', result)`, а тільки `writeln(result)`.
- Робити будь-яку затримку (байдуже, чи через зайві, непотрібні для читання реальних вхідних даних `read`-и або `readln`-и, чи через `repeat until keypressed`, чи ще якось — заборонені всі способи).
- Писати `uses crt`

Недотримання даних вимог автоматично призводить до неможливості перевірити програму автоматичними засобами, а отже — до оцінки в 0 балів.

Задача C «Сума квадратів» (250 балів)

Для заданого натурального числа *N*, визначити, скількома різними способами можна розкласти його в суму двох точних додатних квадратів.
Іншими словами: для заданого *N*, з'ясувати, скільки є різних способів подати його як $N = x^2 + y^2$, причому *x* та *y* являють собою цілі строго додатні числа, а розкладення, в яких значення *x* та *y* лише обміняні місцями, вважаються однаковими.

Вхідні дані — натуральне число *N* — слід прочитати зі стандартного входу (клавіатури).

Результат — знайдену кількість способів — слід вивести на стандартний вихід (екран).

Приклади введення і виведення:

Стандартний вхід (клавіатура)	Стандартний вихід (екран)	Примітка
16	0	Розкласти у суму *додатних* точних квадратів неможливо
10	1	Єдине розкладення $10 = 1^2 + 3^2$
4225	4	Чотири різні розкладення: $4225 = 16^2 + 63^2 = 25^2 + 60^2 = 33^2 + 56^2 = 39^2 + 52^2$

Обмеження *N* — натуральне, $1 \le N \le 123456789$.
100 балів (з 250) припадатиме на тести, в яких $1 \le N \le 1234$.
Ще 50 балів — на тести, в яких $12345 \le N \le 123456$.
Решта 100 балів — на тести, в яких $12345678 \le N \le 123456789$.
Здати потрібно *одну* програму, а не окремі для трьох випадків; різні обмеження вводяться виключно для того, щоб дати приблизне уявлення, скільки балів можна отримати, розв'язавши задачу не повністю.

Обмеження на час роботи програми: 1 секунда

У програмах категорично заборонено:

- Писати будь-які повідомлення, як-то наприклад "Vvedit' N". Програма повинна ***відразу***, без будь-яких запрошень, читати вказані в умові задачі вхідні дані. Знайшовши результат, программа повинна вивести ***тільки*** його. Наприклад, повинно бути ні в якому разі ***не*** `writeln('Kilkist sposobiv ', result)`, а тільки `writeln(result)`.
- Робити будь-яку затримку (байдуже, чи через зайві, непотрібні для читання реальних вхідних даних `read`-и або `readln`-и, чи через `repeat until keypressed`, чи ще якось — заборонені всі способи).
- Писати `uses crt`

Недотримання даних вимог автоматично призводить до неможливості перевірити програму автоматичними засобами, а отже — до оцінки в 0 балів.

Задача D «Дільники» (250 балів)

Для натурального числа *N*, виведіть у порядку зростання всі його різні натуральні дільники.

Вхідні дані слід прочитати зі стандартного входу (клавіатури). Це буде єдине натуральне число *N*.

Результат — послідовність усіх різних натуральних дільників, у порядку зростання — слід вивести на стандартний вихід (екран). Виводити обов'язково в один рядок, розділяючи пробілами.

Приклади введення і виведення:

Стандартний вхід (клавіатура)	Стандартний вихід (екран)
9	1 3 9
120	1 2 3 4 5 6 8 10 12 15 20 24 30 40 60 120

Обмеження N — натуральне, $1 \le N \le 1234567891011$.

120 балів (з 250) припадатиме на тести, в яких $1 \le N \le 4321$.

Решта 130 балів — на тести, в яких $12345678 \le N \le 1234567891011$.

Здати потрібно *одну* програму, а не окремі для двох випадків; різні обмеження вводяться виключно для того, щоб дати приблизне уявлення, скільки балів можна отримати, розв'язавши задачу не повністю.

Обмеження на час роботи програми: 2 секунди

У програмах категорично заборонено:

- Писати будь-які повідомлення, як-то наприклад "Vvedit' N". Програма повинна ***відразу***, без будь-яких запрошень, читати вказані в умові задачі вхідні дані. Знайшовши результат, программа повинна вивести ***тільки*** його. Наприклад, повинно бути ні в якому разі ***не*** `writeln('Kilkist sposobiv ', result)`, а тільки `writeln(result)`.
- Робити будь-яку затримку (байдуже, чи через зайві, непотрібні для читання реальних вхідних даних `read`-и або `readln`-и, чи через `repeat until keypressed`, чи ще якось — заборонені всі способи).
- Писати `uses crt`

Недотримання даних вимог автоматично призводить до неможливості перевірити програму автоматичними засобами, а отже — до оцінки в 0 балів.