<https://www.e-olymp.com/uk/problems/678>

Працівники

На заводі кожна з **N** деталей може бути обробленою на одному з двох верстатів: **A** або **B**. Кожна деталь має порядковий номер від **1** до **N**. До обробки деталі поступають послідовно, у відповідності зі своїми номерами. Кількість деталей завжди парна.

Існують правила, за якими визначається чи можна обробляти деталь на певному верстаті.

1. Якщо на поточний момент на верстаті **B** була оброблена така ж кількість деталей, як і на верстаті **A**, то наступна деталь повинна бути оброблена на верстаті **A**.
2. У підсумку на кожному з верстатів повинно бути оброблено однакову кількість деталей.

Скільки існує людей, стільки і думок. Кожен із працівників цього заводу запропонував свою послідовність обробки деталей, причому всі пропозиції виявилися різними, але такими, що задовольняють правилам **1** і **2**.

Напишіть програму, що за інформацією про кількість деталей N визначає максимальну можливу кількість працівників заводу.

**Вхідні дані**

Одне парне число **N** (**2** ≤ **N** ≤ **28**) - кількість деталей, яку необхідно обробити.

**Вихідні дані**

Вивести одне ціле число - максимальну можливу кількість працівників заводу.

Ліміт часу **1** секунда

Ліміт використання пам'яті **64** MiB

**Вхідні дані**

4

**Вихідні дані**

2

<https://www.e-olymp.com/uk/problems/2479>

Баланс дужок

Є рядок, що містить дужки **( )** та **[ ]**. Дужковий вираз вважається правильним, якщо:

* він є порожнім
* якщо **A** та **B** правильні, то **AB** правильно
* якщо **A** правильно, то (**A**) та [**A**] правильні

Напишіть програму, яка за вхідним рядком, що містить дужковий вираз, визначить його правильність. Довжина рядка не більша за **128**символів.

**Вхідні дані**

Перший рядок містить кількість тестів **n**. Кожний з наступних **n** рядків містить вираз, який складається із дужок **( )** та **[ ]**.

**Вихідні дані**

Для кожного тесту вивести в окремому рядку "**Yes**", якщо вираз є правильним і "**No**" інакше.

Ліміт часу **1** секунда

Ліміт використання пам'яті **122.49** MiB

**Вхідні дані #1**

3

([])

(([()])))

([()[]()])()

**Вихідні дані #1**

Yes

No

Yes

# <https://www.e-olymp.com/uk/problems/4680>

# Делители

За заданим натуральним числом **n** обчислити кількість натуральних чисел, які є дільниками **n!** (факторіала числа **n**).

Наприклад, при **n** = **4**, **n!** = **4** \* **3** \* **2** \* **1** = **24**. Це число має наступні дільники: **1**, **2**, **3**, **4**, **6**, **8**, **12**, **24**. Таким чином, шукана кількість становить **8**.

#### Вхідні дані

Одне ціле число **n** (**1** ≤ **n** ≤ **45**).

#### Вихідні дані

Вивести одне ціле число – кількість дільників числа **n!**

Ліміт часу **1** секунда

Ліміт використання пам'яті **122.17** MiB

**Вхідні дані**

4

**Вихідні дані**

8

# <https://www.e-olymp.com/uk/problems/223>

# Еволюція

Під час досліджень, присвячених появі життя на планеті Олімпія, вченими було зроблено декілька сенсаційних відкриттів:

1. Усі живі організми планети походять від однієї бактерії Bitozoria Programulis.

2.Еволюція проходила крок за кроком (за припущенням вчених – під час змін клімату на планеті).

3.На кожному кроці еволюції з кожного виду утворювалися рівно два підвиди, а попередній вид зникав.

4.Якщо вважати появу бактерії Bitozoria Programulis першим кроком еволюції, то нині існуючі живі організми знаходяться на **n**-му кроці.



Щоб не вигадувати назви під час досліджень, вчені пронумерували всі види організмів, що будь-коли існували на планеті. Для цього вони намалювали дерево еволюції із коренем Bitozoria Programulis, яка отримала номер **1**. Далі нумерували види кожного кроку еволюції зліва направо. Таким чином безпосередні підвиди Bitozoria Programulis отримали номери **2** та **3**. Наступними були занумеровані види третього кроку еволюції – підвиди виду **2** отримали номери **4** та **5**, а виду **3 -**номери **6**та **7**, і т.д.

Напишіть програму, яка за номерами двох видів обчислить номер виду їх найближчого спільного предка у дереві еволюції.

**Вхідні дані**

Перший рядок містить кількість етапів еволюції **n**(**1**≤**n**≤**100**), що відбулися на планеті Олімпія до теперішнього часу. Другий та третій рядки містять по одному натуральному числу, що представляють номери видів, для яких потрібно знайти номер їх найближчого спільного предка.

**Вихідні дані**

Вивести одне натуральне число - номер найближчого предка для двох видів.

Ліміт часу **1** секунда

Ліміт використання пам'яті **122.17** MiB

**Вхідні дані #1**

4

15

12

**Вихідні дані #1**

3

**Вхідні дані #2**

18

233016

233008

**Вихідні дані #2**

14563

# <https://www.e-olymp.com/uk/problems/987>

# Цвяхи

На прямій дощечці вбито цвяхи. Довільні два цвяхи можна з'єднати ниточкою. Потрібно з'єднати деякі пари цвяшків ниточками так, щоб до кожного цвяха була прив'язана хоча б одна ниточка, а сумарна довжина усіх ниточок була мінімальною.

#### Вхідні дані

У першому рядку кількість цвяхів **n** (**1** ≤ **n** ≤ **100**). У наступному рядку записано **n** чисел - координати усіх цвяхів (невід'ємні цілі числа, які не перевищують **10000**).

#### Вихідні дані

Вивести мінімальну сумарну довжину всіх ниточок.

Ліміт часу **1** секунда

Ліміт використання пам'яті **122.17** MiB

**Вхідні дані #1**

5

4 10 0 12 2

**Вихідні дані #1**

6

Сначала отсортируем гвоздики по возрастанию координат. Решим следующую подзадачу: найдем минимальную длину веревочек, необходимую для того, чтобы связать первые k гвоздиков согласно условию (обозначим требующуюся для этого длину веревочек ak).

Можно считать, что любая ниточка связывает два соседних гвоздика (иначе ее можно разрезать на несколько частей, связывающих все гвоздики между теми, которые связывала наша ниточка изначально).

Научимся вычислять ak. Заметим, что в оптимальной конфигурации (для первых k гвоздиков) между последним (k-м) и предпоследним ((k-1)-м) гвоздиками ниточка есть всегда, а вот между предпоследним ((k-1)-м) и предпредпоследним ((k-2)-м) она может либо быть, либо не быть. В первом случае первые k-1 гвоздиков удовлетворяют условию задачи, во втором - первые k-2. Значит ak=min(ak-1,ak-2)+lk-1,k, где lk-1,k - расстояние между k-м и k-1-м гвоздиками (в отсортированном массиве). Для удобства вычислений удобно ввести фиктивные первый и нулевой элементы равные 0 и бесконечности соответственно (в реальной программе в роли бесконечности обычно выступает какое-нибудь достаточно большое число, например для данной задачи вполне подойдет 30000). Теперь последовательно заполняя массив a с помощью данной формулы, мы получим верный ответ на поставленную задачу в элементе aN.

Число действий, выполненных данным алгоритмом, пропорционально N.

#include "iostream"

#include "algorithm"

using namespace std;

long int n,a[1000],b[1000];

int main()

{

cin>>n;

for(int i=1;i<=n;i++) cin>>a[i];

/\*for (int i=0;i<n-1;i++)

 for (int j=0;j<n-1;j++)

 if (a[j]>a[j+1])

swap(a[j],a[j+1]);

\*/

sort(a+1,a+n+1);

//for(int i=1;i<=n;i++) cout<<a[i]<<" ";cout<<endl;

b[0]=0;

b[1]=3000000;

for (int i=2; i<=n;i++)

b[i]=min(b[i-1],b[i-2])+a[i]-a[i-1];

//for(int i=0;i<=n;i++) cout<<b[i]<<" ";cout<<endl;

cout<<b[n]<<endl;

return 0;

}