27. HRVATSKA INFORMATIČKA OLIMPIJADA

ZAGREB, 25. TRAVNJA 2021.

Croatian Olympiad in Informatics

April 3rd 2021

Tasks

Task	Time Limit	Memory Limit	Score
Autobahn	1 second	512 MiB	100
Cigle	1 second	512 MiB	100
Izvanzemaljci	2.5 seconds	512 MiB	100
MalnaRISC	1 second	512 MiB	100
Total			400

Task Autobahn

There are N people testing their racing cars on notorious *autobahn* where limits do not exist. In this task however limits do exist. So we kindly ask you to restrain yourself from submitting exponential complexity solutions.

Person i came to *autobahn* at the beginning of minute l_i, paid for t_i minutes of stay and left at the end of minute r_i. Unfortunately some stayed for longer than that they have paid for. Administration of *autobahn* decided not to be very harsh and charge them only for those extra minutes in which there were at least K people on *autobahn*.

In a rush of generosity, administration decided to introduce *happy hour* i.e. interval of continuous X minutes for which they won't be paying extra charges. They picked *happy hour* so that the sum of extra charges that won't be paid is maximal possible. Determine that sum.

Input

First line contains integers N, K and X $(K \leq N)$ from task description.

Next N lines contain integers l_i, t_i and r_i $(l_i \leq r_i)$ from task description.

Output

Print the required sum in a single line.

Scoring

Subtask	Points	Constraints
1	20	$1 \leq N, K, X, l_i, t_i, r_i \leq 100$
2	30	$1 \leq N, K, X, l_i, t_i, r_i \leq 1\,000$
3	50	$1 \leq N \leq 10^5$, $1 \leq X, l_i, t_i, r_i \leq 10^9$

Examples

input

```
5 3 4
2 1 4
3 3 7
3 3 8
1 5 7
5 3 8
```

output

```
7
```

input

```
3 2 22
7 16 33
69 14 88
8 10 97
```

output

```
27
```

First sample explanation: *Happy hour* will span from 4th until the 7th minute. Inside that interval, first person should've paid extra for the 4th minute and second, third and fourth person should've paid for 6th and 7th minute.

Task Cigle

In an alternate reality *Earth 616* young Stjepan lives a totally different life. Currently he is enrolled in a brick-crafting course at School of Arts and Design. As every child there, he is obsessed with patterns. For example, his homework requires him to build a brick wall using N bricks. But he will not start building until he is satisfied with his two-dimensional sketch.

On the sketch, every brick can be represented as a rectangle with unit size height and width of size d_i. He chooses the order of bricks beforehand and starts sketching from the bottom-most row.

In the first row he will place some number of bricks going from left to right. In the second row he will be placing bricks from right to left and the first brick in the second row will align with the last brick in the first row (their right edges will align). Next, in the third row he will be placing the bricks again from left to right. The first brick in the third row will align with the last from the second row but this time the left edges. He continues this process until there are no bricks left. He may choose to build wall with arbitrary number of rows.

Stjepan uses super cement so a brick may be placed in the wall so that there is no other brick directly underneath. **Beauty of the wall** is a number of places where 4 bricks touch

For a **given order** of bricks and their respective sizes help find the largest possible *beauty* of the wall.

Input

First line contains an integer N from the task description.

Second line contains N integers d_i from the task description.

Output

Print the largest possible beauty of any wall that can be built.

Scoring

Let M be width of the largest brick. $1 \leq M \leq 5\,000$ will hold unless otherwise specified.

Subtask	Points	Constraints
1	9	$1 \leq N \leq 20$
2	11	$1 \leq N \leq 80$
3	13	$1 \leq N \leq 500$, $1 \leq M \leq 2$
4	15	$1 \leq N \leq 500$
5	52	$1 \leq N \leq 5\,000$

Examples

input

```
6
2 2 2 1 1 2
```

output

```
2
```

input

```
13
9 5 2 8 8 2 5 9 9 7 8 5 10
```

output

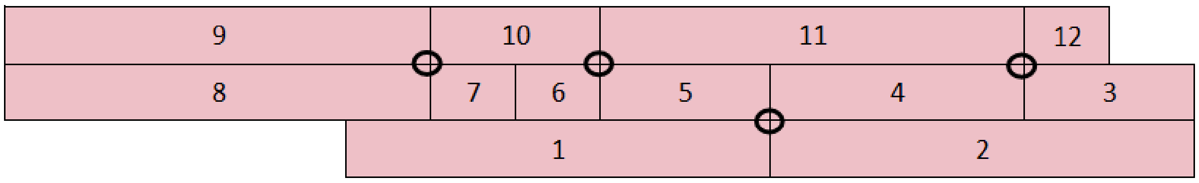
```
5
```

input

```
12
5 5 2 3 2 1 1 5 5 2 5 1
```

output

```
4
```



Wall with beauty 4 for the third example.

Task Izvanzemaljci

It is a well-known fact that a group of Russian scientists discovered an alien civilization back in 2016. Their satellite managed to capture K high-resolution square images that forever changed the course of human history. Today, half a decade later, almost every part of the world has its own space program that investigates aliens. Alas, in Croatia we have decided to tackle some more important problems, which leaves scientific research on the shoulders of few enthusiasts. Vladimir is one of them.

Unfortunately, Vladimir doesn't have enough resources for a spacecraft or an expensive telescope, but he can afford a hot-air balloon and a mobile phone. Instead of an alien civilization, he decided to search for signs of intelligent life on his home planet. Looking down from the balloon, Vladimir notices exactly N people. He decided to capture exactly K square images of average resolution such that each person is seen on exactly one image. Also, he doesn't want any detail to be visible on more than one image. Furthermore, he finds it important that the largest area visible on some picture is as small as possible.

Vladimir is not a great programmer, so he sent you a formal specification and awaits your help.

Formal specification

There are N integer points in the coordinate system. You need to find exactly K disjunct squares that cover all N points. The squares must have sides parallel with the coordinate axes and their vertices should lie on integer coordinates. The area of the largest square needs to be as small as possible.

We say that a square covers a point if the point is fully inside it or lies on its vertices or sides. Two squares are disjunct if their sides doesn't intersect or touch, and neither of the squares is fully contained in the other square.

Input

The first line contains integers N and K from the task description.

The i-th of the next N lines contains integers x_i and y_i ($0 \le |x_i|, |y_i| \le 10^9$) that represent the coordinates of the i-th point (person). All N points will be different.

Output

The i-th of the K lines should contain three integers x_i, y_i ($0 \le |x_i|, |y_i| \le 3 \cdot 10^9$) and l_i ($1 \le l_i \le 2 \cdot 10^9$), that uniquely define the location of the i-th square, such that the point (x_i, y_i) denotes its lower-left vertex, and l_i denotes its side length.

If there are multiple correct solutions, output any of them.

Scoring

Subtask	Points	Constraints
1	5	$1 \le N \le 100\,000$, $K = 1$
2	21	$1 \le N \le 100\,000$, $K = 2$
3	12	$1 \le N \le 12$, $K = 3$
4	30	$1 \le N \le 1\,000$, $K = 3$
5	32	$1 \le N \le 100\,000$, $K = 3$

Examples

input

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

output

```
0 1 2
```

input

```
5 2
1 3
3 1
5 5
5 10
7 7
```

output

```
1 1 4
5 7 3
```

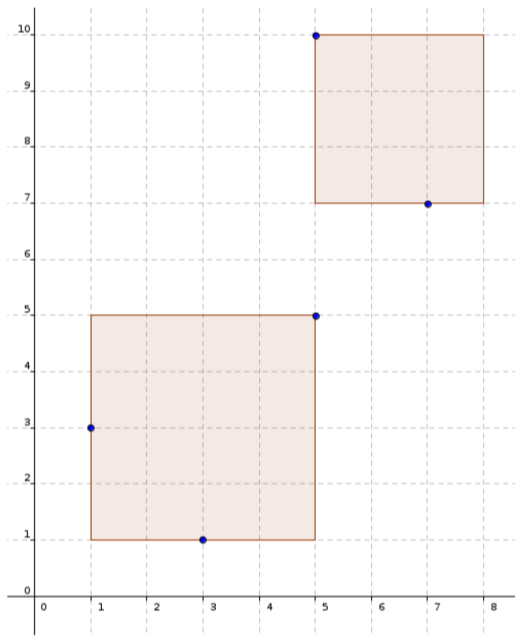
input

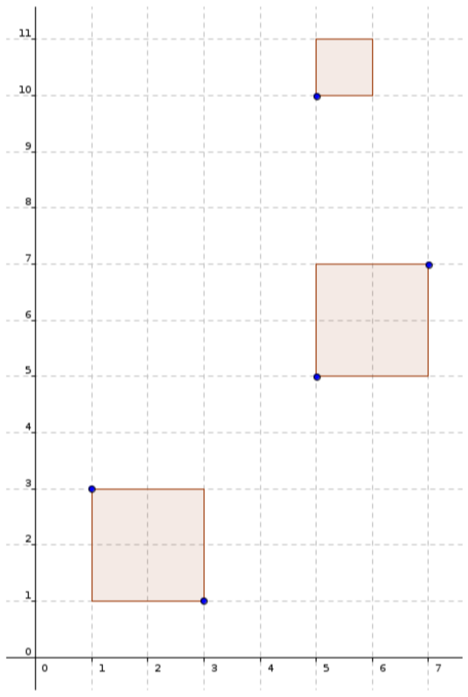
```
5 3
1 3
3 1
5 5
5 10
7 7
```

output

```
1 1 2
5 5 2
5 10 1
```

Explanation of the second and third example:





Task MalnaRISC

It's early in the morning and Croatian IOI team is starting to assemble at the Zagreb airport. The trip is long with the final destination being Singapore with a layover in Amsterdam. Mr. Malar drank the last drop of his grapefruit-based beverage and ordered the team to proceed to the gate. As it usually happens, he disappeared after the security check and somehow managed to show up just a few minutes before boarding.

Olympian 1: Where were you?! I swear you're gonna miss the next flight if you keep doing this.

Mr. Malnar: It's not my fault this time, the security wouldn't let me through. They thought I might be a terrorist.

Olympian 2: A terrorist?! You wouldn't hurt a fly. What happened?

Mr. Malnar: Ah, they found *MalnaRISC* (*Reduced Instruction Set Computer*) and refused to believe me that I am capable of building my own processor. They let me go once I explained how efficient it is at sorting integers.

Olympian 3: I also wouldn't believe you. As a matter of fact, I still don't. What makes your processor so interesting?

Mr. Malnar: You are members of our national IOI team, I shouldn't need to explain anything to you. Here is the documentation, figure it out yourselves.

Olympian 4: Give that to me, I'll solve this year's COI on it using the assembly.

The assembly language for *MalnaRISC* contains a single instruction:

- `CMPSWP` R_i R_j – swaps the values in registers R_i and R_j if $R_i > R_j$ holds.

What's special about *MalnaRISC* is that all instructions written in the same line will execute in parallel during a single nanosecond. Naturally, each register can only be used at most once as an argument in a single line.

It is known that registers R_1, R_2, …, R_N contain some integers. Write an efficient code in assembly that sorts these values in non-descending order.

Input

The only line contains an integer N from the task description.

Output

Output an integer t into the first line denoting the execution time of your program (in nanoseconds).

In the next t lines output the assembly code that sorts the values in the N registers. Each line should contain at least one instruction, and each register should only be mentioned once in a single line. Each instruction needs to be of the form "`CMPSWP` R_i R_j" ($1 \le i, j \le N$), and the instructions in a single line need to be separated by a single space character.

Scoring

Subtask	N	t_1	t_2	t_3	Points
1	8	28	12	6	10
2	13	78	22	10	10
3	16	120	28	10	10
4	32	496	60	15	10
5	53	1378	102	21	10
6	64	2016	124	21	10
7	73	2628	142	28	10
8	82	3321	160	28	10
9	91	4095	178	29	10
10	100	4950	196	30	10

If you have outputted a correct program on some subtask that correctly sorts the values in registers in t nanoseconds, your solutions will be scored according to the following expression:

$$points(t) = \begin{cases} 0 & t > t_1 \\ 1 + \frac{2}{t-t_2} & t_1 \geq t > t_2 \\ 3 + \frac{7(t_2-t+1)}{t_2-t_3} & t_2 \geq t > t_3 \\ 10 & t_3 \geq t \end{cases}$$

The points for each subtask will be rounded to two decimal places. The total scored is obtained by summing these points and rounding that sum in the same manner.

Examples

input

```
2
```

output

```
1
CMPSWP R1 R2
```

input

```
3
```

output

```
3
CMPSWP R1 R2
CMPSWP R1 R3
CMPSWP R2 R3
```

input

```
4
```

output

```
4
CMPSWP R1 R3
CMPSWP R2 R4
CMPSWP R1 R2 CMPSWP R3 R4
CMPSWP R2 R3
```