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 $R_1, R_2, \ldots, R_N$ 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
```