Spice Must Flow

Spice is Love, Spice is Life. And most importantly, Spice must flow. It must be extracted from the scorching sands of Arrakis, under constant threat of giant sand worms. To make the work as efficient as possible, the Duke has tasked you with the creation of a management software.

Write a program that calculates the **total amount** of spice that can be extracted from a source.
The source has a **starting yield**, which indicates how much spice can be mined on the **first day**. After it has been mined for a day, the **yield drops** by 10, meaning on the second day it'll produce 10 less spice than on the first, on the third day 10 less than on the second, and so on (see examples).
A source is considered profitable only while its yield is **at least** 100 – when less than 100 spice is expected in a day, abandon the source.

The mining crew **consumes** 26 spice **every day** at the end of their shift and **an additional** 26 after the mine has been exhausted. Note that the workers cannot consume more spice than there is in storage.

When the operation is complete, print on the console on two separate lines how many days the mine has operated and the total amount of spice extracted.

Input

You will receive a **number**, representing the **starting yield** of the source.

Output

Print on the console on **two separate lines** how many **days** the mine has operated and the **total amount** of spice extracted.

Constraints

- The starting yield will be a positive **integer** within range [0 … 2 147 483 647]

Examples

Input	Output	Explanation
111	2 134	**Day 1** we extract 111 spice and at the end of the shift, the workers consume 26, leaving 85. The yield drops by 10 to 101. **Day 2** we extract 101 spice, the workers consume 26, leaving 75. The total is 160 and the yield has dropped to 91. **Since** the expected yield is less than 100, we abandon the source. The workers take another 26, leaving 134. The mine has operated 2 days.