Calendar Days

Instructions

This challenge involves ordinal dates and string representations of days of the week.

There are two parts to the challenge.

1. Convert ordinal dates into another string-based format.
2. From the output of part 1, create an input for this challenge that returns the same output of part 1.

Endpoint

Create an endpoint `/calendarDays` that accepts a JSON payload over `POST` described below.

Input

Example of said JSON payload:

```
{
    "numbers": [2022, 1, 1, 0, 366]
}
```

The first number in the list represents a year within the range `[2001, 2095]`.

The rest of the numbers, if present, may correspond to the day of the year, starting with 1.

Please ignore any values less than `1` or greater than the number of days of the year.

In the example above, it is simply the 1st day of 2022 (may repeat), when you ignore the invalid values `0` and `366`.

Output

The output (i.e. your answer) to return should be in the shape of:

```
{
    "part1": "weekend,weekday, t     ,alldays,alldays,alldays,alldays,alldays,allday
    "part2": [2022, 1]
}
```

Our evaluation is flexible enough to determine the scoring based on given attributes, more will be explained in the scoring section below.

Part One

Return a 96-character case-sensitive string, with every 8 characters corresponding to the month of the year starting with January.

The 8 characters will be derived in the following order (excluding the quote character `'"'` shown below):

1. If all days of the week are in the list, display `"alldays,"`
2. If only the weekends (Saturday and Sunday) are in the list, display `"weekend,"`
3. If only the weekdays (Monday to Friday) are in the list, display `"weekday,"`
4. Else, display the days of the week in the format `"mtwtfss,"`, using an empty character `' '` for absent days

Each criterion must appear at least once (and only that criteria) to trigger any of the first three outputs. For example, if there is only one weekend appearing for a given month, case `2` will be the output for that month. If the first fifteen days of the month is in the list, then case `1` will be the output for that month because it covers all seven days of the week.

Examples

Input	Explanation	Expected Output
`[2022]`	No days	`"       ,       ,       ,       ,       ,`
`[2022, 1]`	First day of 2022 = Saturday	`"     s ,       ,       ,       ,       ,`
`[2022, 1, 2, 8, 9, 15, 16, 22, 23, 29, 30]`	All January 2022 weekends	`"weekend,       ,       ,       ,       ,`
`[2022, 38, 39,`	38th to 42nd days of	`"       ,weekday`

Input	Explanation	Expected Output
`40, 42, 41]`	2022 = First full weekday in February	`"       ,weekday,       ,       ,       ,`
`[2022, -1, 0, 1, 2, 60, 38, 40, 39, 42, 41, 91 ... 334, 444 ... 999]`		`"weekend,weekday, t     ,alldays,alldays,allday`

Part Two

From part 1's answer, derive a new year by adding `2001` to the (zero-based) index of the first whitespace character `' '`.

Then, return a list of numbers, starting with the derived year, such that using it as part 1's input will also result in the expected part 1's answer.

Examples

Input
`"       ,       ,       ,       ,       ,       ,       ,       ,`
`"m      , t     ,weekend,       ,       ,       ,       ,       ,`

Scoring

Refresher on the output:

```
{
    "part1": "weekend,weekday, t     ,alldays,alldays,alldays,alldays,alldays,allday
    "part2": [2022, 1]
}
```

If we encounter any error parsing the response, the score is `0`.

Part 1

If the string is not exactly 96 characters containing 12 commas `","`, the score is `0`.

Else, the string will be split by the 12 commas into 13 values, i.e. the 12 months and a trailing empty `""` value.

If more than 7 of these values matches the expected output (case-sensitive), the score will be `(2 * count) - 1` for a maximum of `25` (`2 * 13 - 1`).

Else the score is `0`.

Part 2

If the list of numbers is empty, or if the first element is not the expected year, the score is `0`.

Else, this list will be mapped to a part 1 answer and evaluated based on part 1's criteria above.

This means that the maximum score for part 2 is also `25`.

Scoring Summary

The score range for part 1 is `[0, 25]` and that for part 2 is `[0, 25]`.

There will be four evaluation attempts, summing to a maximum of `200` before that raw score is halved (rounding down), then multiplied by the challenge weightage.

Example

See here.