Div and Span

Maya is teaching Alex about HyperText Markup Language (HTML). Alex is confused about *div* and *span* tags, so Maya decides to reduce the concept to a simpler visual by representing *div* tags as square brackets and *span* tags as parentheses. In other words, she represents `<div>` as `[`, `</div>` as `]`, `<span>` as `(`, and `</span>` as `)`.

We use the following rules to determine the validity of a configuration:

1. The empty string (""), a single pair of square brackets `[]`, and single pair of parentheses `()` are all considered to be valid configurations. We can express each of these as V.
2. A valid configuration, V, which *does not contain* square brackets is called a *normal* configuration, which we can express as N. For example, `()` is *normal*.
3. Two or more consecutive valid configurations (e.g., VV) also constitute a valid configuration. For example, `[][]`, `()()`, and `[]()[]` are all valid.
4. The following configurations are also valid: `[V]`, `(N)`.
   For example, `[[(())]()]`, `((()))`, and `[(())[]]` are all valid; however, `([])` is *not valid* (you cannot nest a *div* tag inside a *span* tag).

Given some number of distinguishable square brackets, X, and some number of distinguishable parentheses, Y, how many valid configurations can we build using the above rules? As this answer can be very large, print it modulo $10^9 + 7$.

Input Format

The first line contains a single positive integer, T, denoting the number of test cases.
Each of the T subsequent lines contains 2 positive space-separated integers describing the respective values of X (the number of distinguishable square brackets) and Y (the number of distinguishable parentheses).

Constraints

- $1 \le T \le 1000$
- $1 \le X \le 10^5$
- $1 \le Y \le 200$

Output Format

For each test case, print the number of different valid configurations modulo $10^9 + 7$.

Sample Input

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

Sample Output

```
3
18
160
```

Explanation

For the first test case, $X = 1$, $Y = 1$, these are 3 valid configurations:

1. `[]()`
2. `()[]`
3. `[()]`

Thus, we print the result of $3 \ \% \ (10^9 + 7)$ on the first line.

For the second test case, if brackets and parentheses were not distinguishable, we would have only 9 different configurations:

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

However, they *are* distinguishable, so `[](1)(2)` is not same as `[](2)(1)` (where `(1)` and `(2)` are the respective first and second pairs of parentheses. For this reason, we have 18 possible configurations and print the result of $18 \ \% \ (10^9 + 7)$ on a new line.