Problem D: Pebbles

You're given an unlimited number of pebbles to distribute across an N x N game board (N drawn from [3, 15]), where each square on the board contains some positive point value between 10 and 99, inclusive. A 6 x 6 board might look like this:

33	74	26	55	79	54
67	56	91	72	44	32
44	64	22	91	29	61
61	32	76	50	50	32
81	65	56	38	96	36
38	78	50	92	90	75

The player distributes pebbles across the board so that:

- At most one pebble resides in any given square.
- No two pebbles are placed on adjacent squares. Two squares are considered adjacent if they are horizontal, vertical, or even diagonal neighbors. There's no board wrap, so 44 and 61 of row three aren't neighbors. Neither are 33 and 75 nor 55 and 92.

The goal is to maximize the number of points claimed by your placement of pebbles.

Write a program that reads in a sequence of boards from an input file and prints to stdout the maximum number of points attainable by an optimal pebble placement for each.

Input (from file d.in)

Each board is expressed as a series of lines, where each line is a space-delimited series of numbers. A blank line marks the end of each board (including the last one)

Output (to monitor)

then your program would print the maximum number of points one can get by optimally distributing pebbles while respecting the two rules, which would be this (each output should be printed on a single line and followed with a newline):

Sample Input

```
71 24 95 56 54
85 50 74 94 28
92 96 23 71 10
23 61 31 30 46
64 33 32 95 89

78 78 11 55 20 11
98 54 81 43 39 97
12 15 79 99 58 10
13 79 83 65 34 17
85 59 61 12 58 97
40 63 97 85 66 90

33 49 78 79 30 16 34 88 54 39 26
80 21 32 71 89 63 39 52 90 14 89
49 66 33 19 45 61 31 29 84 98 58
36 53 35 33 88 90 19 23 76 23 76
77 27 25 42 70 36 35 91 17 79 43
33 85 33 59 47 46 63 75 98 96 55
75 88 10 57 85 71 34 10 59 84 45
29 34 43 46 75 28 47 63 48 16 19
62 57 91 85 89 70 80 30 19 38 14
61 35 36 20 38 18 89 64 63 88 83
45 46 89 53 83 59 48 45 87 98 21

15 95 24 35 79 35 55 66 91 95 86 87
94 15 84 42 88 83 64 50 22 99 13 32
85 12 43 39 41 23 35 97 54 98 18 85
84 61 77 96 49 38 75 95 16 71 22 14
18 72 97 94 43 18 59 78 33 80 68 59
26 94 78 87 78 92 59 83 26 88 91 91
34 84 53 98 83 49 60 11 55 17 51 75
29 80 14 79 15 18 94 39 69 24 93 41
66 64 88 82 21 56 16 41 57 74 51 79
49 15 59 21 37 27 78 41 38 82 19 62
54 91 47 29 38 67 52 92 81 99 11 27
31 62 32 97 42 93 43 79 88 44 54 48
```

Sample Output

```
572
683
2096
2755
```