Lab 2.3.9 The riddle (a bit of a tricky one)

Objectives

- inspire the student to look for non-simple solutions;
- prove that the first option is not always the best one;
- relax the student.

Scenario

We think that now is a good time for a riddle. Of course, we don't expect you to solve the riddle. We want you to write a program that will solve it.

Imagine a square built out of tiles. The size of its side (measured in the number of tiles) is always odd. For example, this is a square with a side size equal to 3:

```
+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+
```

Now we're going to fill it with numbers. We always start from the very center tile and place a "1" in it. The next number is obviously "2" and we place it above the center tile. The next number will be "3" and we put it to the right of "2".

Have you already guessed what we're doing? Yes, we're going to form a right-handed helix!

This is what our square will eventually look like:

```
+---+---+---+
| 9 | 2 | 3 |
+---+---+---+
| 8 | 1 | 4 |
+---+---+---+
| 7 | 6 | 5 |
+---+---+---+
```

Here's what we expect from you: write a code that accepts one integer number (it must be odd), treat it as the size of our "squared helix" and find the number located in the bottom right tile (in our example it's 5).

Hint: there are many ways to find the answer – some of them are better (shorter and faster), while some are worse (longer and slower). The fact that you can use a loop doesn't mean that you have to use it. Think twice, use paper and pencil, do not rush. Better solutions usually take time to find.

And, of course, test your code using the data we've provided.

Example input

```
3
```

Example output

```
5
```

Example input

```
5
```

Example output

```
17
```

Example input

```
11
```

Example output

```
101
```

Example input

```
11111
```

Example output

```
123432101
```