Algorithm: Max Squares in Right Isosceles Triangle (Reduction Technique)

Name & Purpose

Computes the maximum number of non-overlapping 2×2 squares that fit inside a right isosceles triangle with base B, where one square side is parallel to the base.

Inputs

- **b**: int, the base (and height) of the right isosceles triangle.
    - Constraint: $b \geq 1$

Outputs

- **int**: the maximum count of 2×2 squares that fit; range $[0, \infty)$

Preconditions

- b is a positive integer
- The triangle is right isosceles with the base as the shortest side

Postconditions

- Returns a non-negative integer count of squares

High-Level Idea

Use reduction (divide and conquer): peel off the bottom horizontal strip of height 2 from the triangle, count how many 2×2 squares fit there (n = b//2 - 1), then recursively solve the remaining smaller similar triangle with base b-2.

Base case: if b < 4, no squares fit, return 0.

Pseudocode

```
function max_squares(b):
  if b < 4:
    return 0
  n ← b // 2 - 1                   // squares in the current bottom strip
  return n + max_squares(b - 2)      // add squares from the reduced triangle
```

Complexity

- **Time**: O(b) – b/2 recursive calls, each O(1). Dominant term: b/2.

- **Space**: O(b) – recursion depth is b/2; call stack stores O(b) frames.

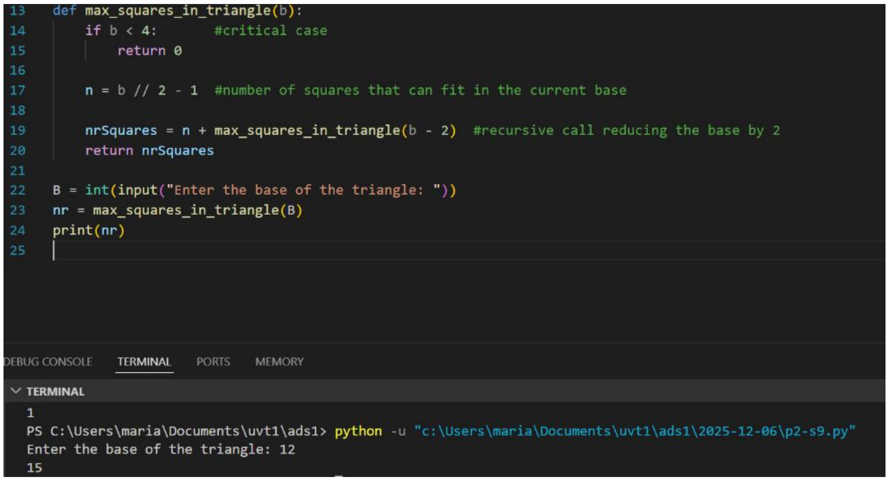
Correctness Sketch

Invariant: each strip of height 2 at level k contains exactly (k-1) non-overlapping 2×2 squares (for k ≥ 2). By induction: base case (b < 4) is correct; inductive step removes one valid strip and recurses, so total = n + f(b-2) is correct.

Edge Cases

- **b = 1, 2, 3**: fewer than 4 units → return 0 (no 2×2 square fits)
- **b = 4, 5**: exactly one strip fits → return 1
- **Non-integer input**: convert to int (e.g., 5.7 → 5)
- **Negative b**: return 0

Example



```python
def max_squares_in_triangle(b):
    if b < 4:        #critical case
        return 0

    n = b // 2 - 1  #number of squares that can fit in the current base

    nrSquares = n + max_squares_in_triangle(b - 2)  #recursive call reducing the base by 2
    return nrSquares

B = int(input("Enter the base of the triangle: "))
nr = max_squares_in_triangle(B)
print(nr)
```

```
1
PS C:\Users\maria\Documents\uvt1\ads1> python -u "c:\Users\maria\Documents\uvt1\ads1\2025-12-06\p2-s9.py"
Enter the base of the triangle: 12
15
```