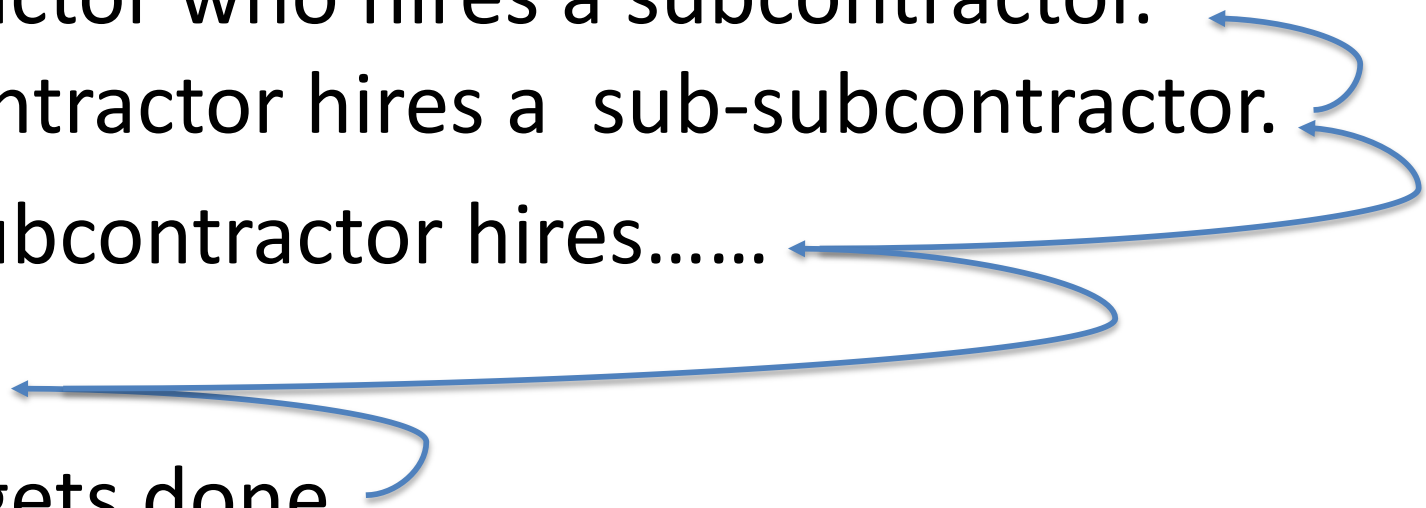
Recursion

- See slide titled “Recursion” for more ☺
- Process in which the result of each repetition is dependent upon the result of the next repetition
- Simplifies program structure at the cost of function calls
- Sesquipedalian
  - A person who might use words like “Sesquipedalian”

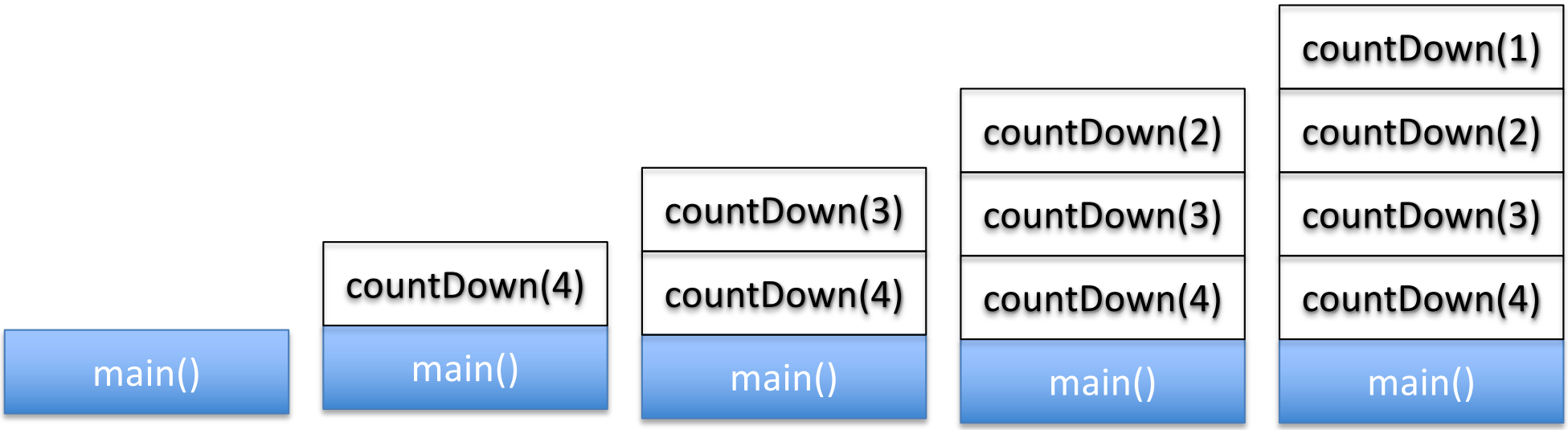
- Real life examples
  - MC Escher drawings
  - Fractals

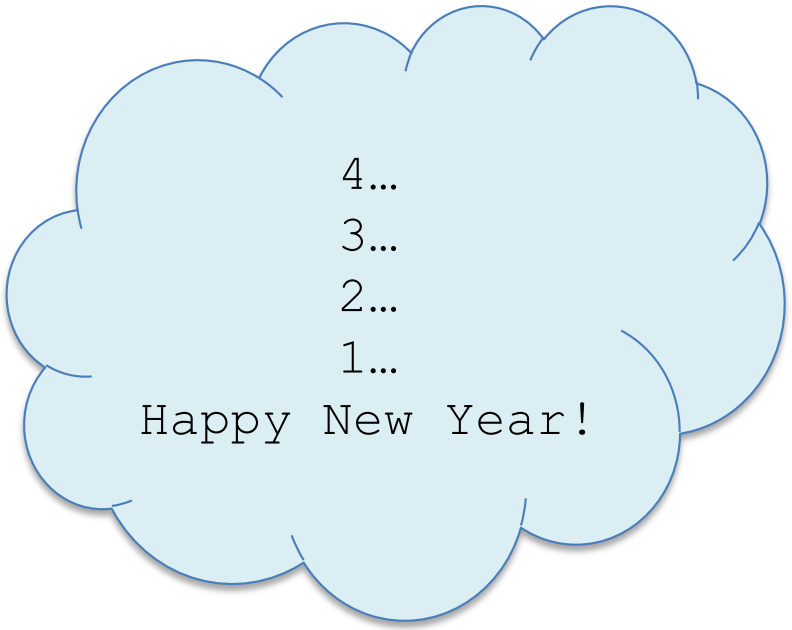
Contractor who hires a subcontractor.
Subcontractor hires a sub-subcontractor.
Sub-subcontractor hires……
……
work gets done



```java
package IntroConcepts;

public class BallDrop {

    public void countDown(int leadTime) {
        System.out.println(leadTime + "...");
        if (leadTime > 1)
            countDown(leadTime - 1);
        else
            System.out.println("Happy New Year!");
    }


}

```

```java
package IntroConcepts;

import student.TestCase;

public class BallDropTest extends TestCase{
    BallDrop counter;

    public void setUp() {
        counter = new BallDrop();
    }

    public void testCountDown(){
        //could capture output and compare it to expected
        counter.countDown(10);
    }

}
```


main()
countDown(4)
main()
countDown(3)
countDown(4)
main()
countDown(2)
countDown(3)
countDown(4)
main()
countDown(1)
countDown(2)
countDown(3)
countDown(4)
main()


4…
3…
2…
1…
Happy New Year!

Key Components

- **Base Case:**
  - where recursion stops!
- **Recursive Case:**
  - Gets you one step closer to the base case
  - A leap of faith!
- How do you set 99 bottles of Snapple off of the wall?

A Recursive Method

- Calls itself
- Takes input
- Has logic based on input
  - Leads to recursive case
  - Leads to base case [stop]
- Probably contains if statement
- Probably does not have a loop