Pragmatisk programmering

Rekursjon

Hva er egentlig rekursjon?

- Enkelt forklart: En *rekursiv metode* er en metode som kaller på seg selv (ref. IN1150, for de som har det)
- Vi trenger to ting:
  1. Et basistilfelle: Noe som må oppnås før metoden kan slutte å kalle på seg selv
  2. Et (eller flere) rekursjonssteg: Gjør noe endringer for å komme nærmere basistilfellet, og kall på metoden igjen med disse endringene

Metoder som kan gjøres iterativt kan også gjøres rekursivt (og motsatt)

```java
public class Rekursjon {
    public static void main(String[] args) {
        tellNed( n: 5);
    }

    public static void tellNed(int n) {  2 usages
        if (n < 0) return;
        System.out.println(n);
        tellNed( n: n - 1);
    }
}
```

```java
public class Iterasjon {
    public static void main(String[] args) {
        tellNed( n: 5);
    }

    public static void tellNed(int n) {  1 usage
        while (n >= 0) {
            System.out.println(n);
            n--;
        }
    }
}
```

Call stack

- Når man kaller på en metode, legges metodekallet på en *call stack*
  - Repetisjon: Hva er en *stack*?
- En rekursiv metode vil legge metodekallene sine på call stacken, og først når vi har nådd basistilfellet vil vi gå “tilbake” i stacken
  - Alle elementene som er lagt på, poppes da fra stacken
  - Det siste kallet poppes først, og det første kallet poppes sit

```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack

Output



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack

tellOpp(5)

Output



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack

tellOpp(5)

Output



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack

tellOpp(5)

tellOpp(4)

Output



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack

tellOpp(5)

tellOpp(4)

Output



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack

tellOpp(5)

tellOpp(4)

tellOpp(3)

Output



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack

tellOpp(5)

tellOpp(4)

tellOpp(3)

Output



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack

tellOpp(5)

tellOpp(4)

tellOpp(3)

tellOpp(2)

Output



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack

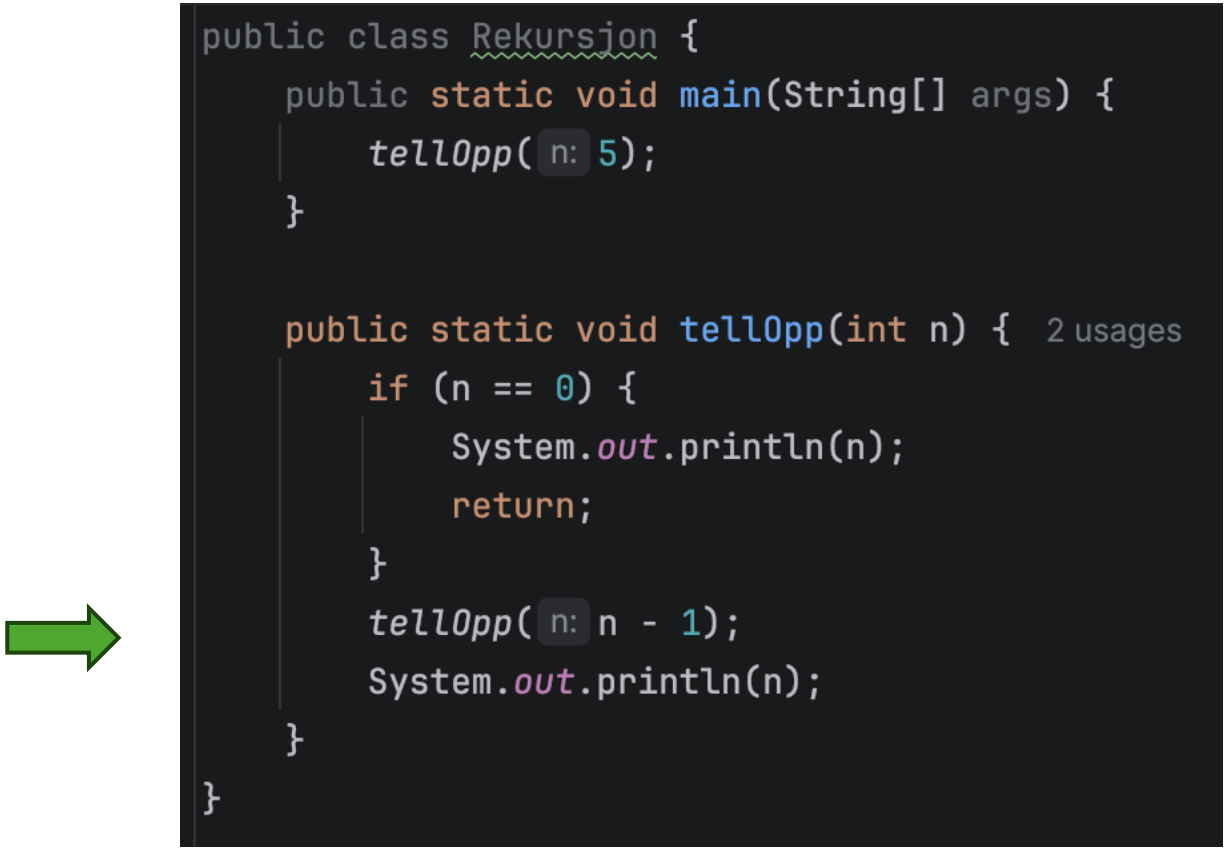
tellOpp(5)

tellOpp(4)

tellOpp(3)

tellOpp(2)

Output



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack

tellOpp(5)

tellOpp(4)

tellOpp(3)

tellOpp(2)

tellOpp(1)

Output



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack

tellOpp(5)

tellOpp(4)

tellOpp(3)

tellOpp(2)

tellOpp(1)

Output



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack

tellOpp(5)

tellOpp(4)

tellOpp(3)

tellOpp(2)

tellOpp(1)

tellOpp(0)

Output



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack

tellOpp(5)

tellOpp(4)

tellOpp(3)

tellOpp(2)

tellOpp(1)

tellOpp(0)

Output



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

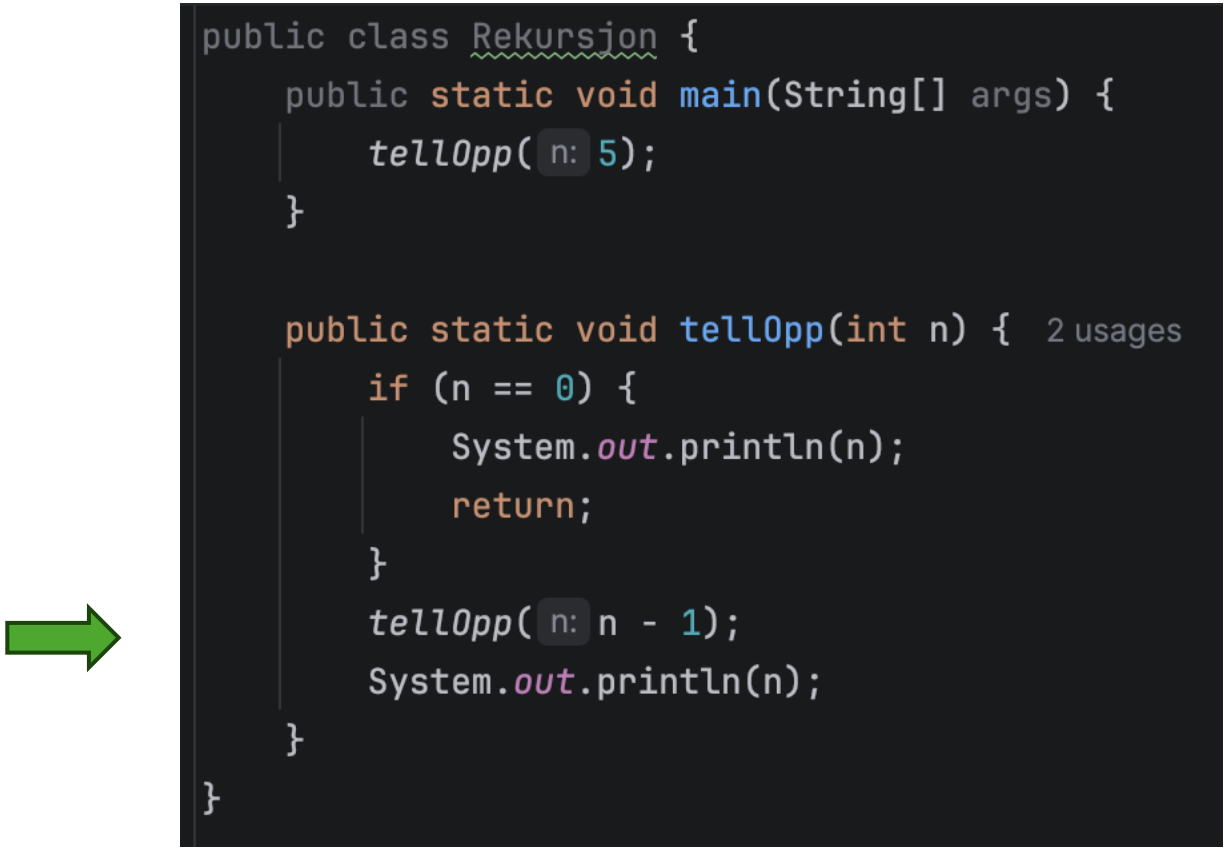
Stack	Output
tellOpp(5)	0
tellOpp(4)	
tellOpp(3)	
tellOpp(2)	
tellOpp(1)	
tellOpp(0)	



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

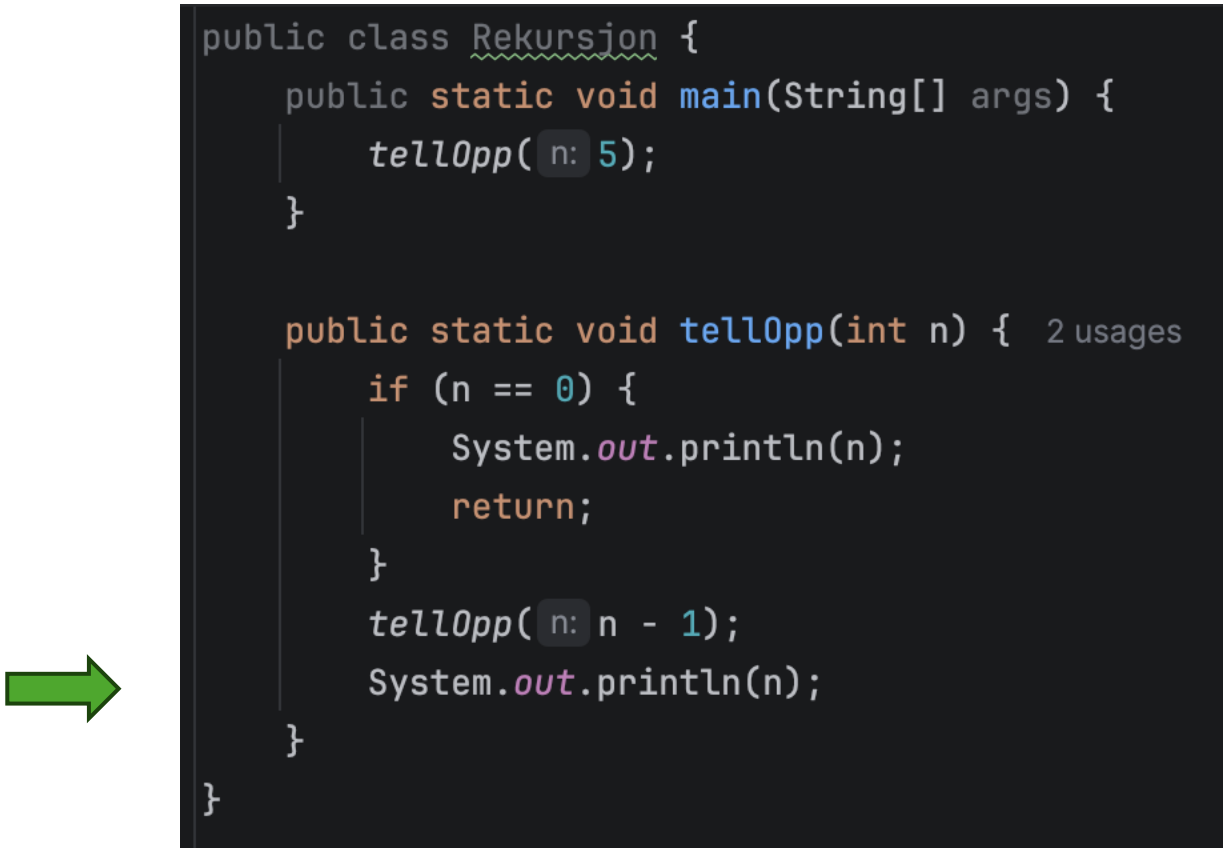
Stack	Output
tellOpp(5)	0
tellOpp(4)	
tellOpp(3)	
tellOpp(2)	
tellOpp(1)	
tellOpp(0)	



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

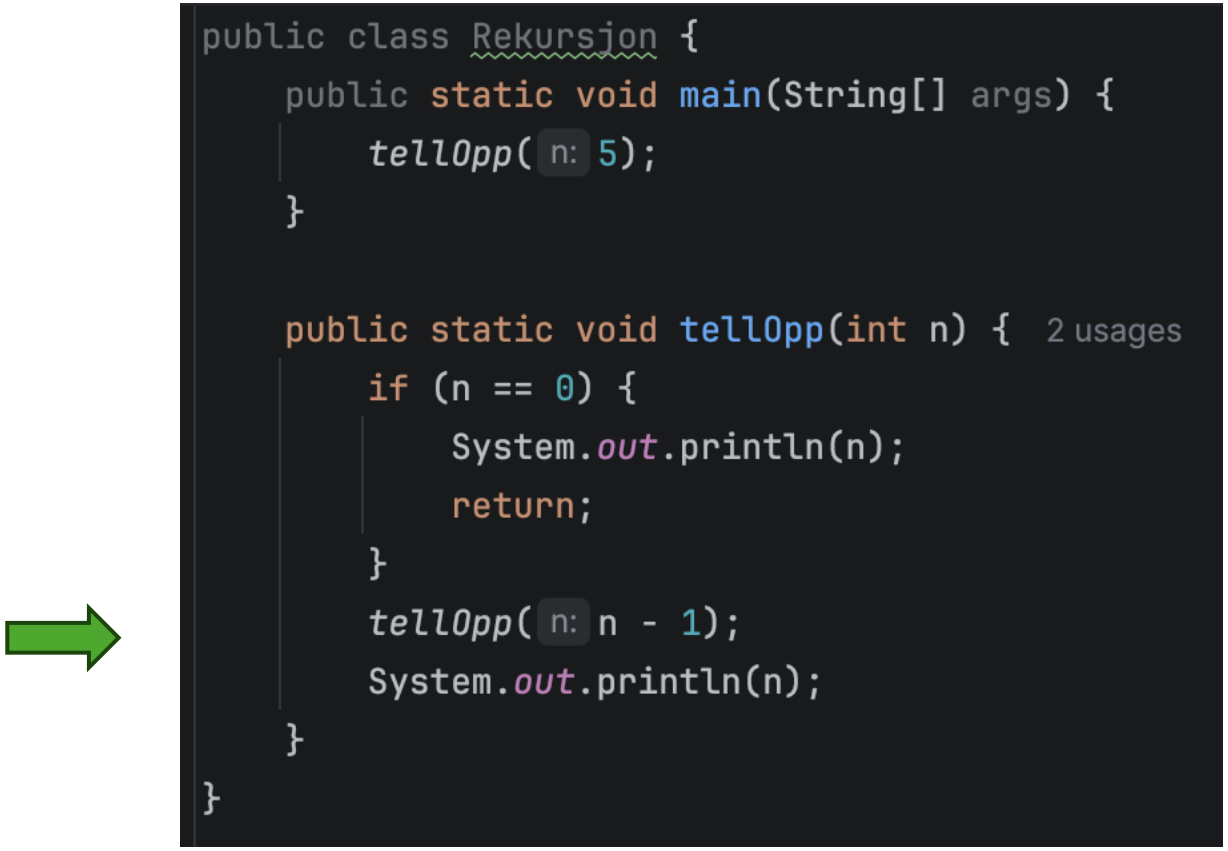
Stack	Output
tellOpp(5)	0
tellOpp(4)	
tellOpp(3)	
tellOpp(2)	
tellOpp(1)	



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack	Output
tellOpp(5)	0
tellOpp(4)	1
tellOpp(3)	
tellOpp(2)	
tellOpp(1)	



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack	Output
tellOpp(5)	0
tellOpp(4)	1
tellOpp(3)	
tellOpp(2)	



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

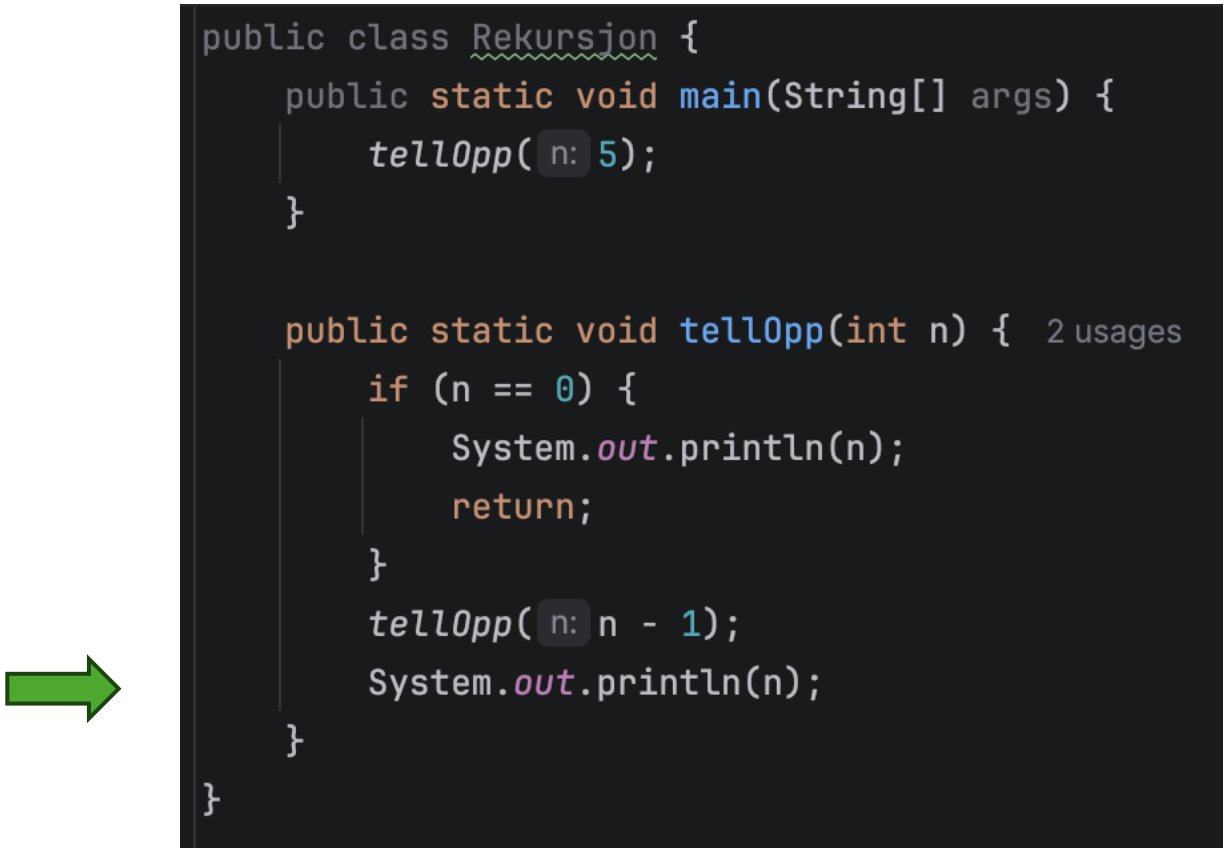
Stack	Output
tellOpp(5)	0
tellOpp(4)	1
tellOpp(3)	2
tellOpp(2)	

```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```



Stack	Output
tellOpp(5)	0
tellOpp(4)	1
tellOpp(3)	2



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack	Output
tellOpp(5)	0
tellOpp(4)	1
tellOpp(3)	2
	3

```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```



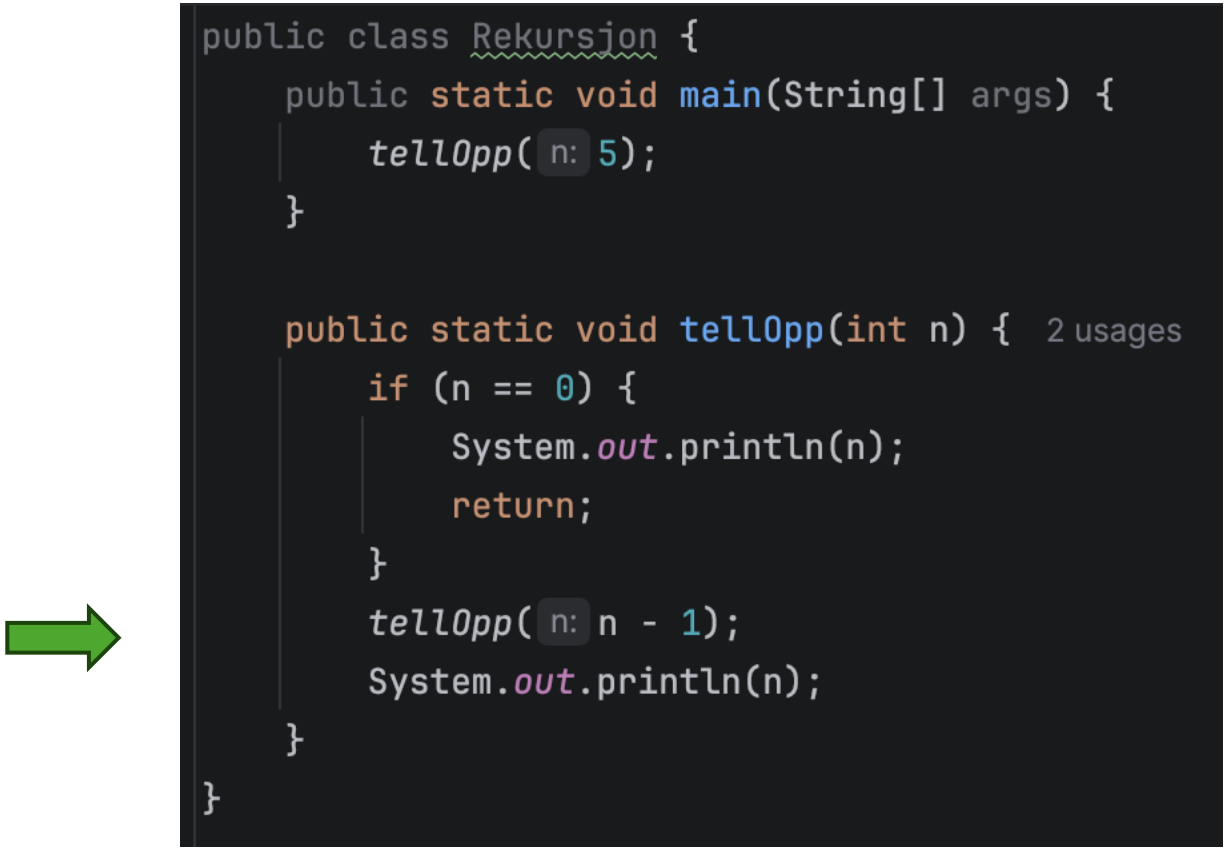
Stack	Output
tellOpp(5)	0
tellOpp(4)	1
	2
	3



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack	Output
tellOpp(5)	0
tellOpp(4)	1
	2
	3
	4



```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack	Output
tellOpp(5)	0
	1
	2
	3
	4

```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```



Stack	Output
tellOpp(5)	0
	1
	2
	3
	4
	5

```java
public class Rekursjon {
    public static void main(String[] args) {
        tellOpp( n: 5);
    }

    public static void tellOpp(int n) {  2 usages
        if (n == 0) {
            System.out.println(n);
            return;
        }
        tellOpp( n: n - 1);
        System.out.println(n);
    }
}
```

Stack

Output

0

1

2

3

4

5