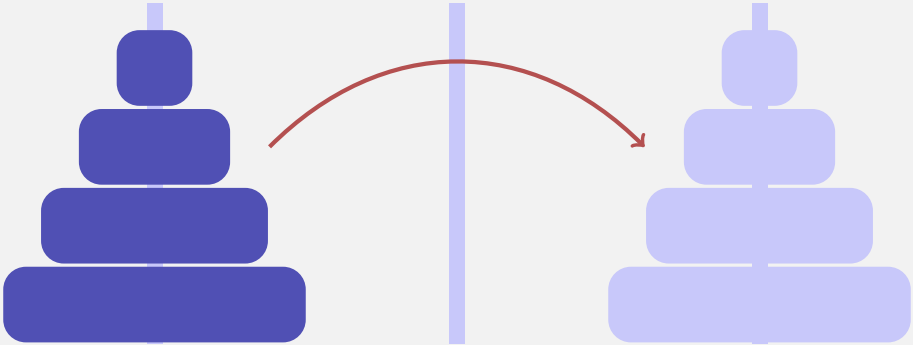
Experiment: The Towers of Hanoi

left middle right

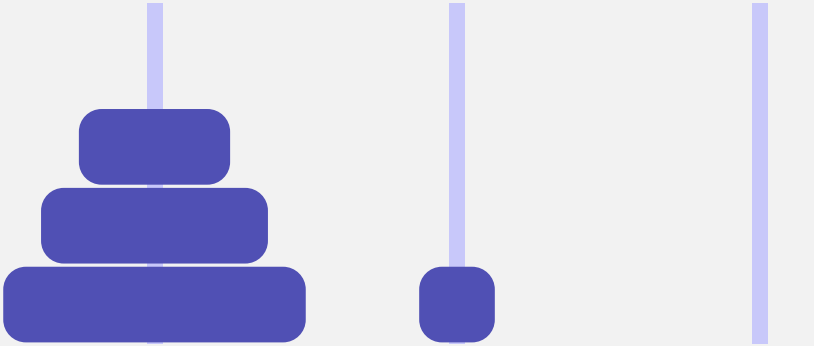
Experiment: The Towers of Hanoi



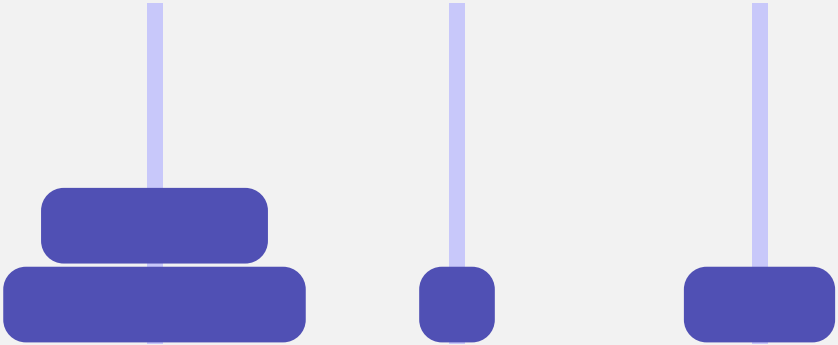
Die Türme von Hanoi - So gehts!

left middle right

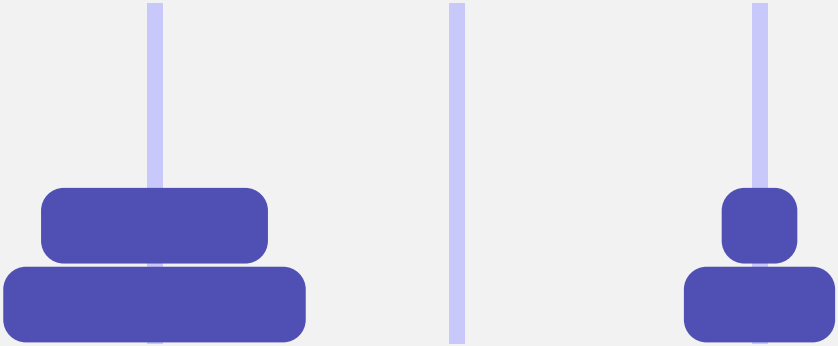
Die Türme von Hanoi - So gehts!



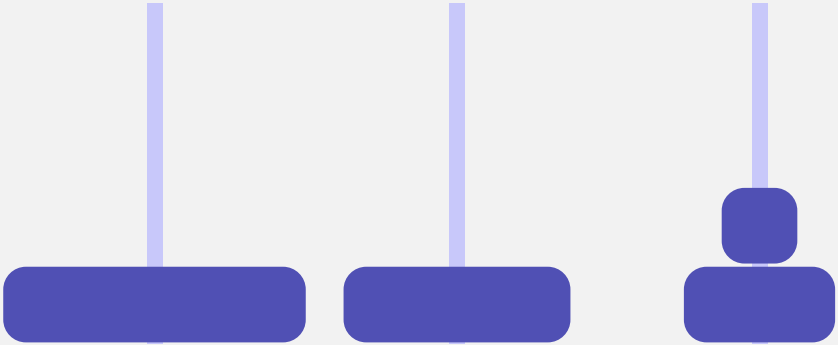
Die Türme von Hanoi - So gehts!



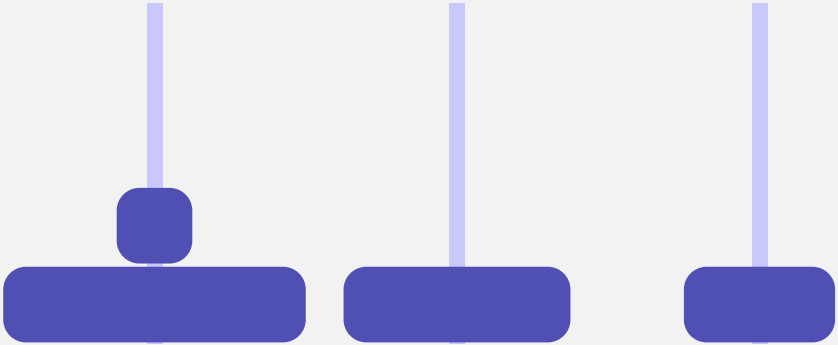
Die Türme von Hanoi - So gehts!



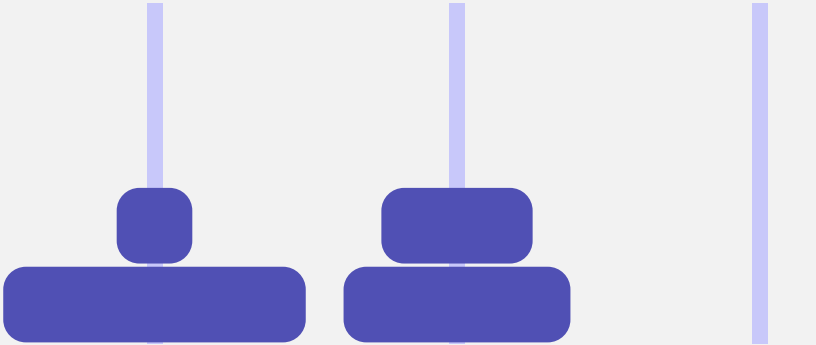
Die Türme von Hanoi - So gehts!



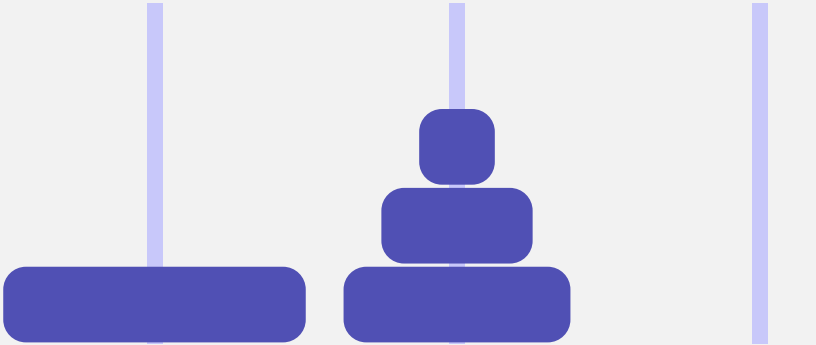
Die Türme von Hanoi - So gehts!



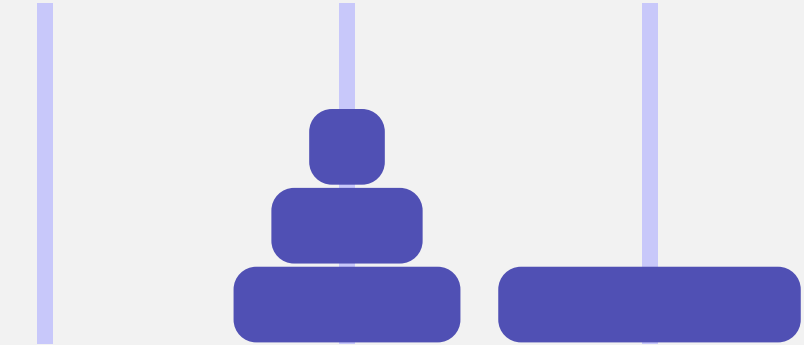
Die Türme von Hanoi - So gehts!



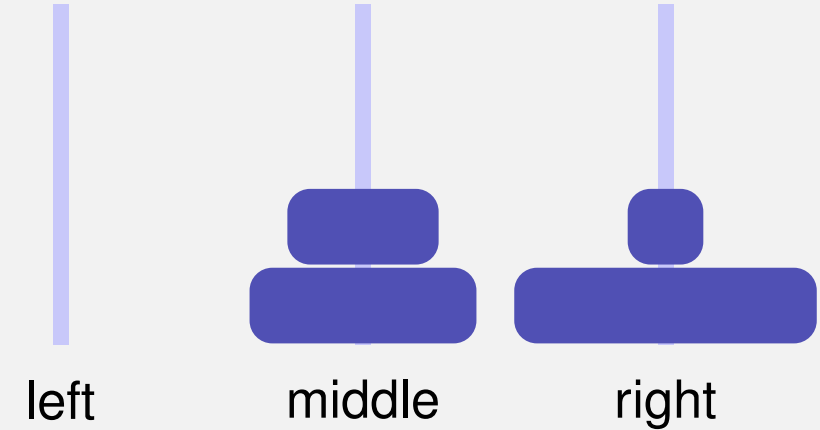
Die Türme von Hanoi - So gehts!



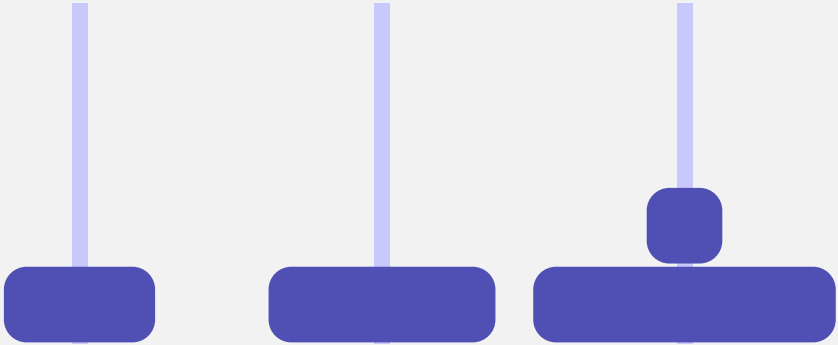
Die Türme von Hanoi - So gehts!



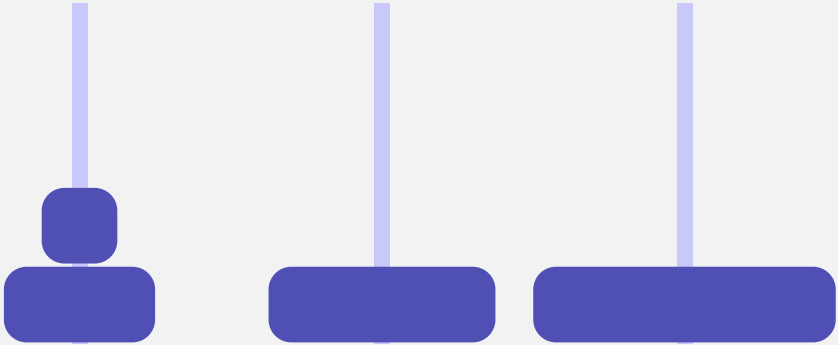
Die Türme von Hanoi - So gehts!



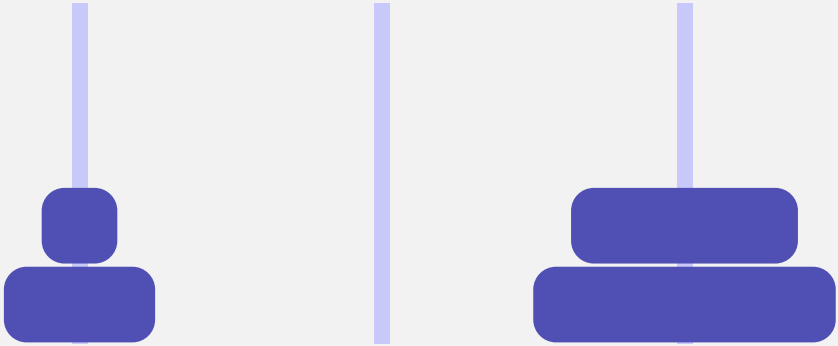
Die Türme von Hanoi - So gehts!



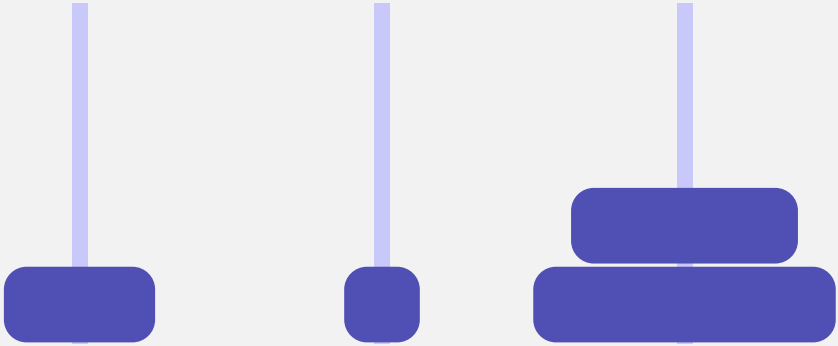
Die Türme von Hanoi - So gehts!



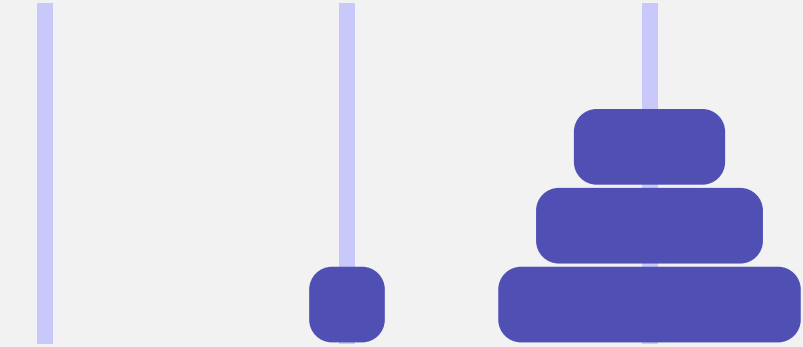
Die Türme von Hanoi - So gehts!



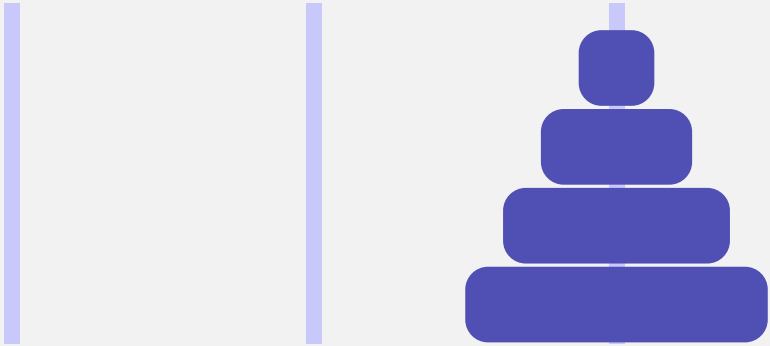
Die Türme von Hanoi - So gehts!



Die Türme von Hanoi - So gehts!

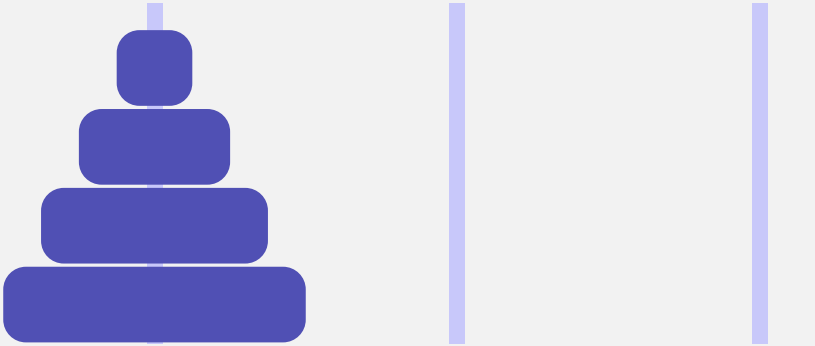


Die Türme von Hanoi - So gehts!

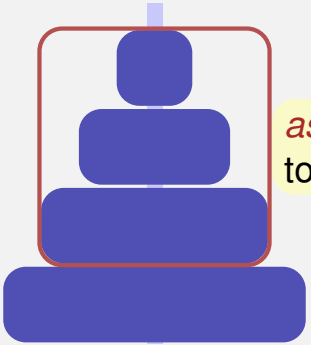


left middle right

The Towers of Hanoi – Recursive Approach



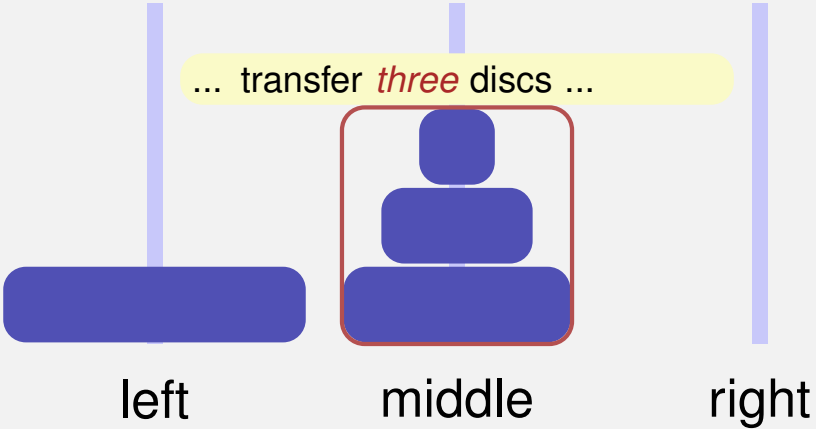
The Towers of Hanoi – Recursive Approach



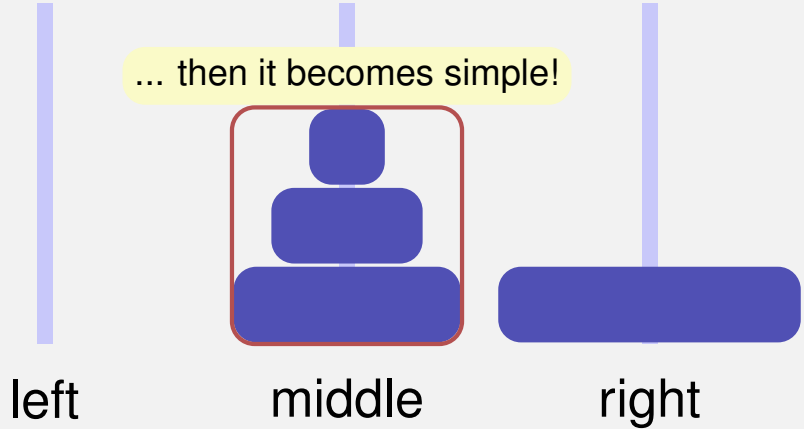
assume we knew how to ...

left middle right

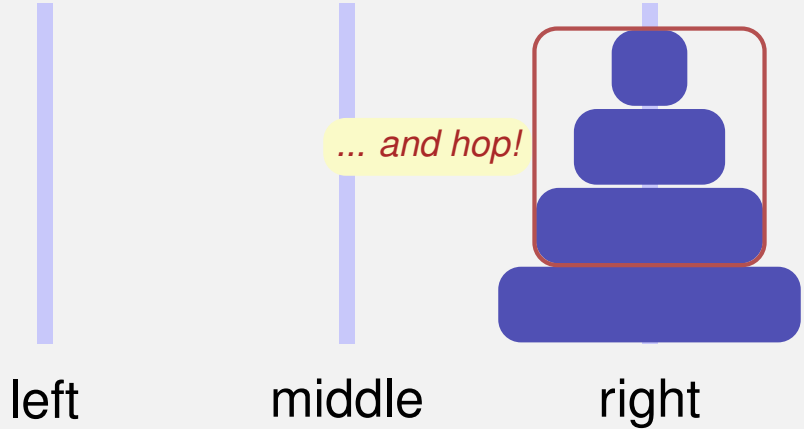
The Towers of Hanoi – Recursive Approach



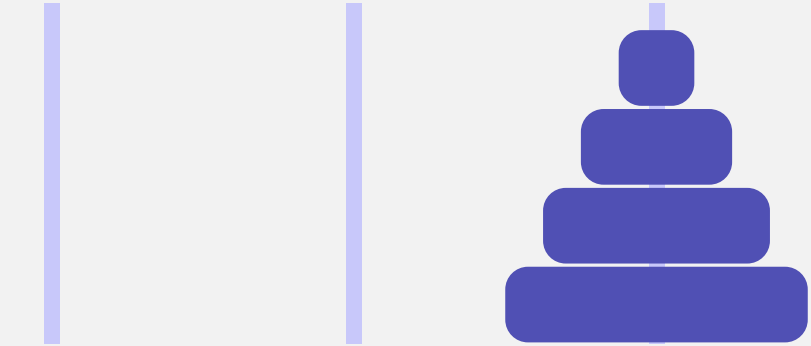
The Towers of Hanoi – Recursive Approach



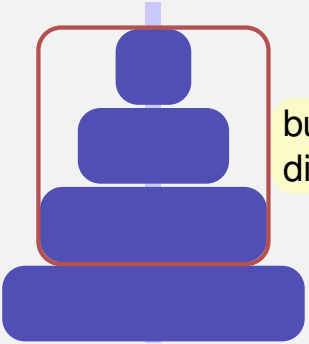
The Towers of Hanoi – Recursive Approach



The Towers of Hanoi – Recursive Approach



The Towers of Hanoi – Recursive Approach



but *how* can we move three discs?

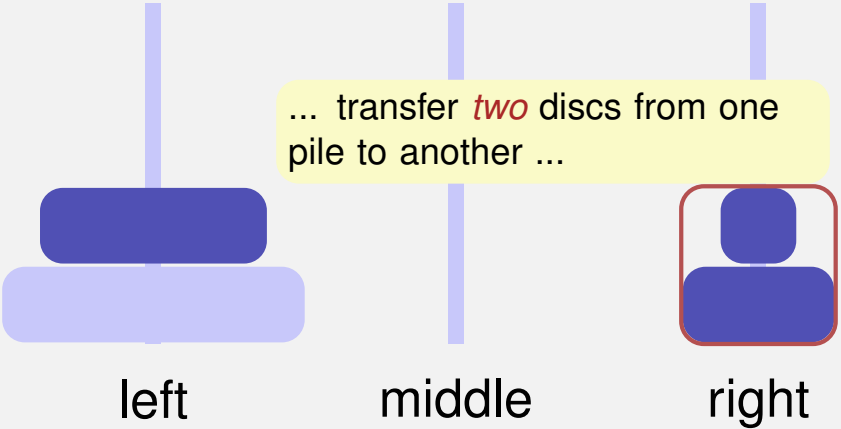
left middle right

The Towers of Hanoi – Recursive Approach

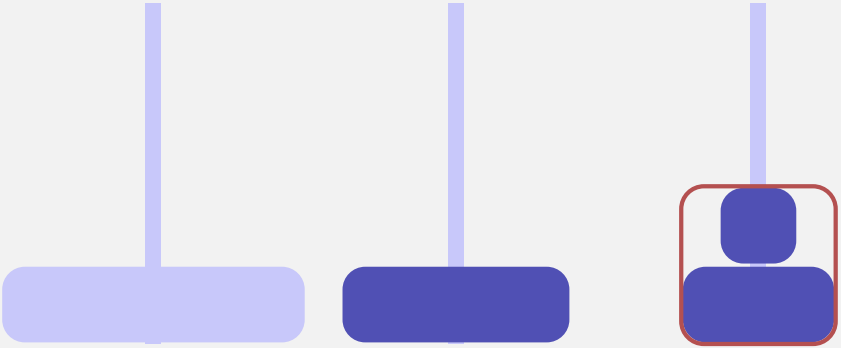
assume we knew how to ...

left middle right

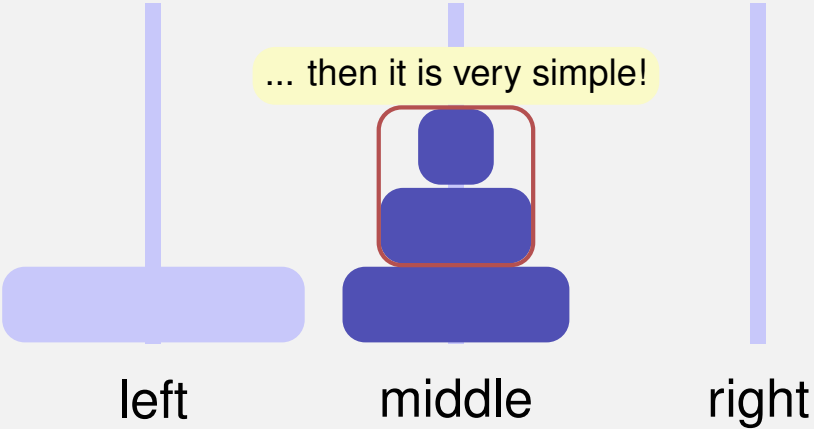
The Towers of Hanoi – Recursive Approach



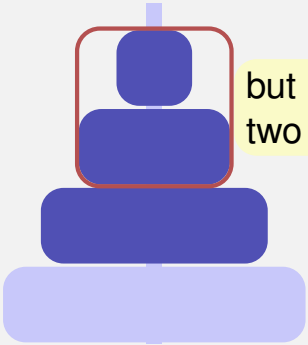
The Towers of Hanoi – Recursive Approach



The Towers of Hanoi – Recursive Approach



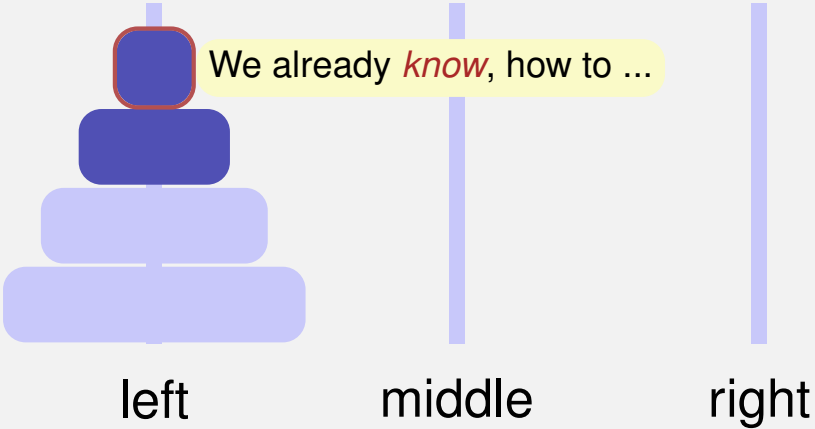
The Towers of Hanoi – Recursive Approach



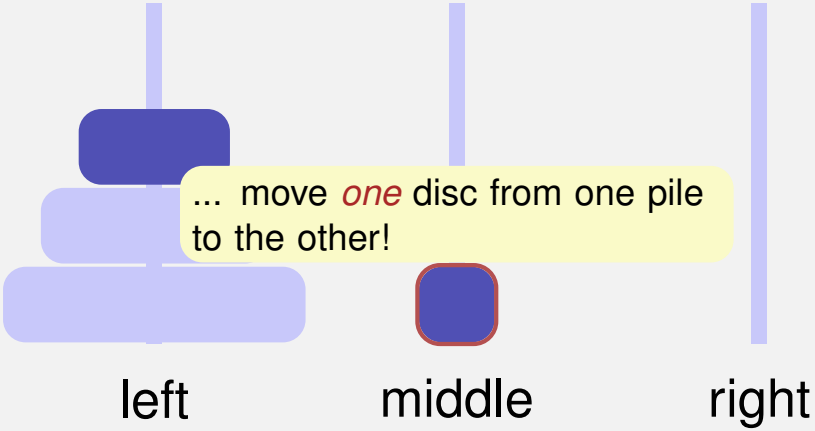
but *how* can we transfer two discs?

left middle right

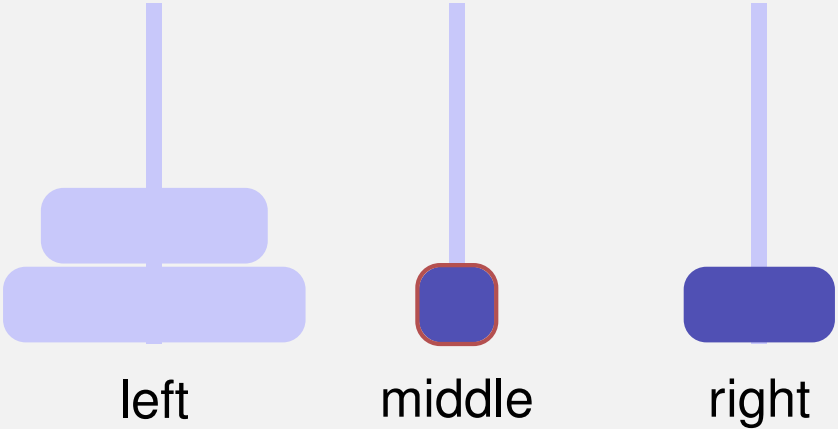
The Towers of Hanoi – Recursive Approach



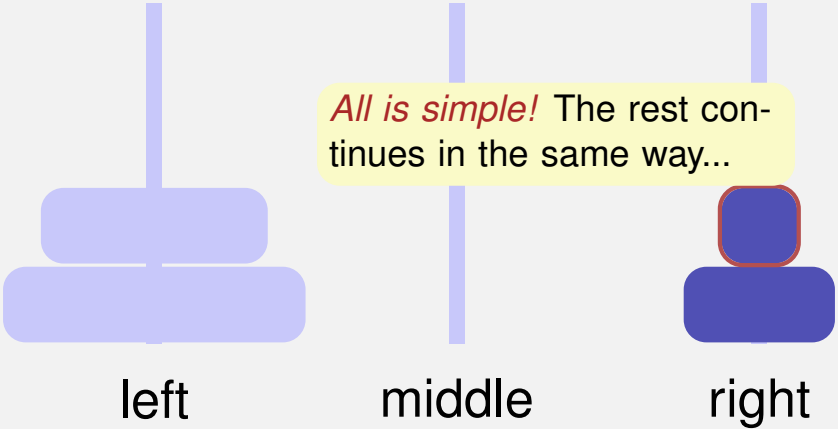
The Towers of Hanoi – Recursive Approach



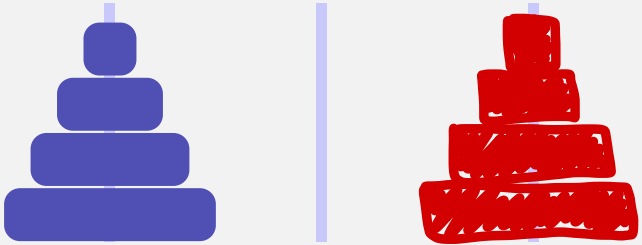
The Towers of Hanoi – Recursive Approach



The Towers of Hanoi – Recursive Approach



The Towers of Hanoi – Code



Move 4 discs from `left` to `right` with auxiliary staple `middle`:

src aux dst

```
move(4,"left","middle","right")
```

The Towers of Hanoi – Code

(string)

`move`$(n,\ src,\ aux,\ dst)\quad \Rightarrow$

1. Move the top $n-1$ discs from src to aux with auxiliary staple dst:
   `move`$(n-1,\ src,\ dst,\ aux)$`;`
2. Move 1 disc from src to dst
   `move(1,` $src,\ aux,\ dst$`);`
3. Move the top $n-1$ discs from aux to dst with auxiliary staple src:
   `move`$(n-1,\ aux,\ src,\ dst)$`;`

else

The Towers of Hanoi – Code

main():
move(n, "left", "middle", "right")

```
void move(int n, const string &src, const string &aux, const string &dst){
    if (n == 1) {
        // base case (‘move’ the disc)
        std::cout << src << " --> " << dst << std::endl;
    } else {
        // recursive case



    }
}
```

Base case

The Towers of Hanoi – Code

```
void move(int n, const string &src, const string &aux, const string &dst){
    if (n == 1) {
        // base case (‘move’ the disc)
        std::cout << src << " --> " << dst << std::endl;
    } else {
        // recursive case
        move(n-1, src, dst, aux);


    }
}
```

The Towers of Hanoi – Code

```
void move(int n, const string &src, const string &aux, const string &dst){
    if (n == 1) {
        // base case (‘move’ the disc)
        std::cout << src << " --> " << dst << std::endl;
    } else {
        // recursive case
        move(n-1, src, dst, aux);
        move(1, src, aux, dst);

    }
}
```

The Towers of Hanoi – Code

```
void move(int n, const string &src, const string &aux, const string &dst){
    if (n == 1) {
        // base case (‘move’ the disc)
        std::cout << src << " --> " << dst << std::endl;
    } else {
        // recursive case
        move(n-1, src, dst, aux);
        move(1, src, aux, dst);
        move(n-1, aux, src, dst);
    }
}
```

The Towers of Hanoi – Code

```
void move(int n, const string &src, const string &aux, const string &dst){
    if (n == 1) {
        // base case ('move' the disc)
        std::cout << src << " --> " << dst << std::endl;
    } else {
        // recursive case
        move(n-1, src, dst, aux);
        move(1, src, aux, dst);
        move(n-1, aux, src, dst);
    }
}
int main() {
    move(4, "left", "middle", "right");
    return 0;
}
```

The Towers of Hanoi – Code Alternative

```
void move(int n, const string &src, const string &aux, const string &dst){
    // base case
    if (n == 0) return;

    // recursive case
    move(n-1, src, dst, aux);
    std::cout << src << " --> " << dst << "\n";
    move(n-1, aux, src, dst);
}


int main() {
    move(4, "left", "middle", "right");
    return 0;
}
```