```
 1:int gcd(int m, int n)
 2:{
 3:    int x;
 4:
 5:    /* base case: check for 0 */
 6:    if (n == 0) return(m);
 7:
 8:    /* recurse */
 9:    x = gcd(n, m % n);
10:
11:    /* done! */
12:    return(x);
13:}
```

```
14:
15:int main(void)
16:{
17:    int n;
18:
19:    n = gcd(6,4);
20:    printf("GCD of 4 and 6 is %d\n", n);
21:    return(0);
22:}
```

Initial call to gcd: gcd(m ← 4, n ← 6)

```
 1: int gcd(int m, int n)
 2: {
 3:     int x;
 4:
 5:     /* base case: check for 0 */
 6:     if (n == 0) return(m);
 7:
 8:     /* recurse */
 9:     x = gcd(n, m % n);
10:
11:     /* done! */
12:     return(x);
13: }
```

gcd(4, 6): return to main, line 19
m = 4, n = 6

gcd(m ← 4, n ← 6) :

6: condition false, so skip

9: call gcd(6, 4)

```
 1: int gcd(int m, int n)
 2: {
 3:     int x;
 4:
 5:     /* base case: check for 0 */
 6:     if (n == 0) return(m);
 7:
 8:     /* recurse */
 9:     x = gcd(n, m % n);
10:
11:     /* done! */
12:     return(x);
13: }
```


gcd(6, 4): return to line 9, purple arrow
m = 6, n = 4

gcd(4, 6): return to main, line 19
m = 4, n = 6

gcd(m ← 6, n ← 4) :

6: condition false, so skip

9: call gcd(4, 2)

```
 1: int gcd(int m, int n)
 2: {
 3:     int x;
 4:
 5:     /* base case: check for 0 */
 6:     if (n == 0) return(m);
 7:
 8:     /* recurse */
 9:     x =    gcd(n, m % n);
10:
11:     /* done! */
12:     return(x);
13: }
```

gcd(4, 2): return to line 9, red arrow m = 4, n = 2
gcd(6, 4): return to line 9, purple arrow m = 6, n = 4
gcd(4, 6): return to main, line 19 m = 4, n = 6

gcd(m ← 4, n ← 2) :
6: condition false, so skip
9: call gcd(2, 0)

```
 1: int gcd(int m, int n)
 2: {
 3:     int x;
 4:
 5:     /* base case: check for 0 */
 6:     if (n == 0) return(m);
 7:
 8:     /* recurse */
 9:     x = gcd(n, m % n);
10:
11:     /* done! */
12:     return(x);
13: }
```

gcd(2, 0): return to line 9, green arrow m = 4, n = 2
gcd(4, 2): return to line 9, red arrow m = 4, n = 2
gcd(6, 4): return to line 9, purple arrow m = 6, n = 4
gcd(4, 6): return to main, line 19 m = 4, n = 6

gcd(m ← 2, n ← 0) :

6: condition true, so return 2

```
 1: int gcd(int m, int n)
 2: {
 3:     int x;
 4:
 5:     /* base case: check for 0 */
 6:     if (n == 0) return(1);
 7:
 8:     /* recurse */
 9:     x = gcd(n, m % n);
10:
11:     /* done! */
12:     return(x);
13: }
```

gcd(2, 0): return to line 9, green arrow m = 4, n = 2; return 2
gcd(4, 2): return to line 9, red arrow m = 4, n = 2
gcd(6, 4): return to line 9, purple arrow m = 6, n = 4
gcd(4, 6): return to main, line 19 m = 4, n = 6

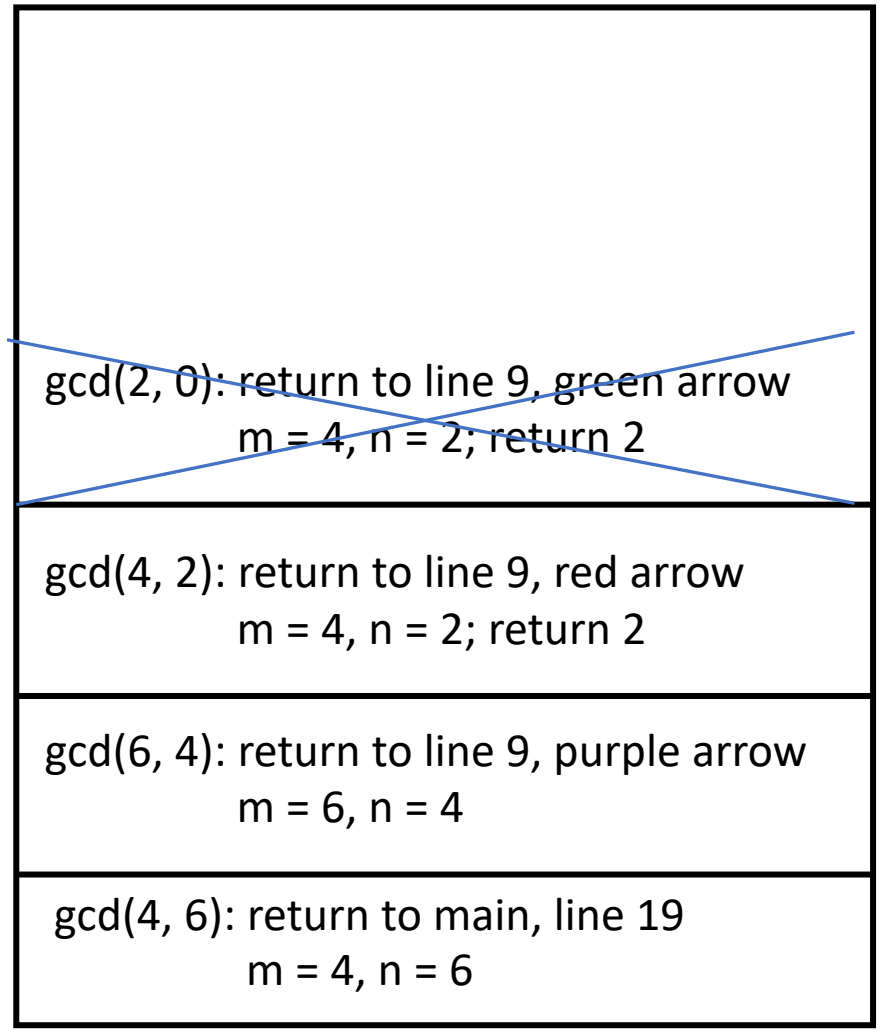
gcd(m ← 4, n ← 2) :

6: condition false, so skip

9: call gcd(2, 0); x = 2

12: return 2

```
 1: int gcd(int m, int n)
 2: {
 3:     int x;
 4:
 5:     /* base case: check for 0 */
 6:     if (n == 0) return(m);
 7:
 8:     /* recurse */
 9:     x = gcd(n, m % n);
10:
11:     /* done! */
12:     return(x);
13: }
```



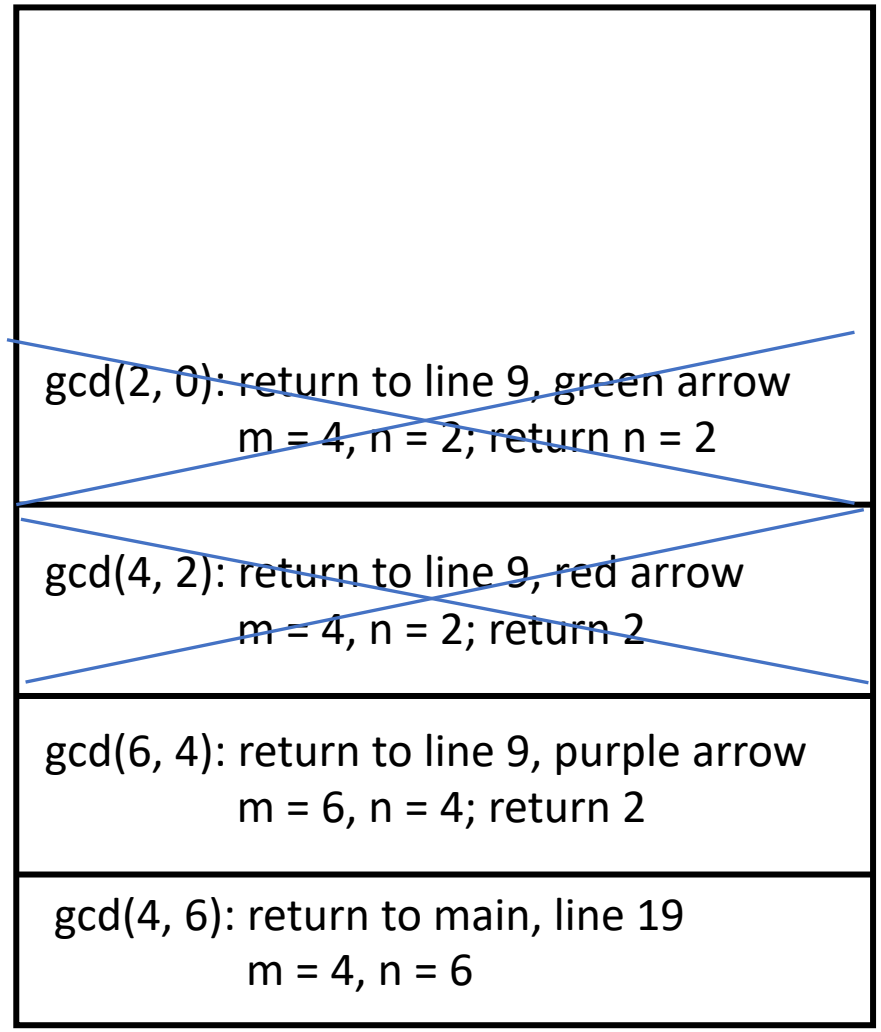
gcd(m ← 6, n ← 4) :

6: condition false, so skip

9: call gcd(4, 2); x = 2

12: return 2

```
 1: int gcd(int m, int n)
 2: {
 3:     int x;
 4:
 5:     /* base case: check for 0 */
 6:     if (n == 0) return(m);
 7:
 8:     /* recurse */
 9:     x =    gcd(n, m % n);
10:
11:     /* done! */
12:     return(x);
13: }
```



gcd(m ← 4, n ← 6) :

6: condition false, so skip

9: call gcd(6, 4); x = 2

12: return 2

```
 1: int gcd(int m, int n)
 2: {
 3:     int x;
 4:
 5:     /* base case: check for 0 */
 6:     if (n == 0) return(m);
 7:
 8:     /* recurse */
 9:     x =    gcd(n, m % n);
10:
11:     /* done! */
12:     return(x);
13: }
```