Loop invariance

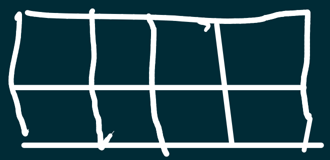
Program verification

```
initialization
while (C) {

    loop body

}
```

a statement S that is true at all times when C is checked.

$m \times n$ chocolate

```
i = 0;
while still not done {
    Pick one bigger than 1 x 1 piece
    Break it into two smaller piece
    i++;
}
```



$mn - 1$

minimum
maximum
no. of breaks necessary?

At all times, there are $i+1$ pieces.

R = 100; G = 101; B = 102;

while (the bag contains balls of at least two colors) {

Pick two balls of different colors;

If the colors are R, G, --R; --G; ++B;

else R, B, --R; --B; ++G;

else G, B, --G; --B; ++R;

}

which color is it? G

R	100	Even	Odd
G	101	Odd	Even
B	102	Even	Odd

At all times,

$$\text{parity}(R) = \text{parity}(B)$$

$$\text{parity}(G) \neq \text{that}$$

$0 \rightarrow$ even parity

```
Fibonacci (n)
    if n is 0, return 0;
    G = 1; H = 0; i = 1;
    while (i < n) {
        F = G + H;
        H = G;
        G = F;
        ++i;
    }
    return G;
```

$G = F_1 = 1$

$H = F_0 = 0$

$F = F_i + F_{i-1} = F_{i+1}$

$H = F_i = F_{(i+1)-1}$

$G = F_{i+1}$

$i = i + 1$

when the loop cond is checked

$G = F_i$, $H = F_{i-1}$

Extended gcd

a, b - two positive integers

$$d = \gcd(a, b)$$
$$= ua + vb \quad \text{for some } u, v \in \mathbb{Z}$$

Compute d, u and v.

$r_0 = a$

$r_1 = b$

$r_0 = q_2 r_1 + r_2$

$r_1 = q_3 r_2 + r_3$

$- - -$

$r_{j-1} = q_{j+1} r_j + r_{j+1}$

$r_j = q_{j+2} r_{j+1} + r_{j+2}$

$r_{j+1} = q_{j+3} r_{j+2}$

$d = \gcd(a, b)$

$= r_{j+1}$

$r_0, r_1, r_2, \ldots$

$u_0, u_1, u_2, \ldots$

$v_0, v_1, v_2, \ldots$

$r_0 = a;$ $u_0 = 1;$ $v_0 = 0;$

$r_1 = b;$ $u_1 = 0;$ $v_1 = 1;$ $i = 1;$

while $(r_i \neq 0)$ {

Euclidean division of r_{i-1} by r_i $\rightarrow q_{i+1}, r_{i+1}$

$r_{i+1} = r_{i-1} - q_{i+1} r_i;$

$u_{i+1} = u_{i-1} - q_{i+1} u_i;$

$v_{i+1} = v_{i-1} - q_{i+1} v_i;$ $++i;$

}

return $(r_{i-1}, u_{i-1}, v_{i-1})$

$\forall i \quad u_i a + v_i b = r_i$

$i = 0, 1$ this holds

$i \geq 1$ invariance holds for $i, i-1$

$$u_{i-1} a + v_{i-1} b = r_{i-1}$$

$$u_i a + v_i b = r_i$$

$r_{i+1} =$

$$r_{i-1} - q_{i+1} r_i = (u_{i-1} a + v_{i-1} b) - q_{i+1}(u_i a + v_i b)$$

$$= (u_{i-1} - q_{i+1} u_i) a + (v_{i-1} - q_{i+1} v_i) b$$

$$= u_{i+1} a + v_{i+1} b.$$

How to improve this algorithm

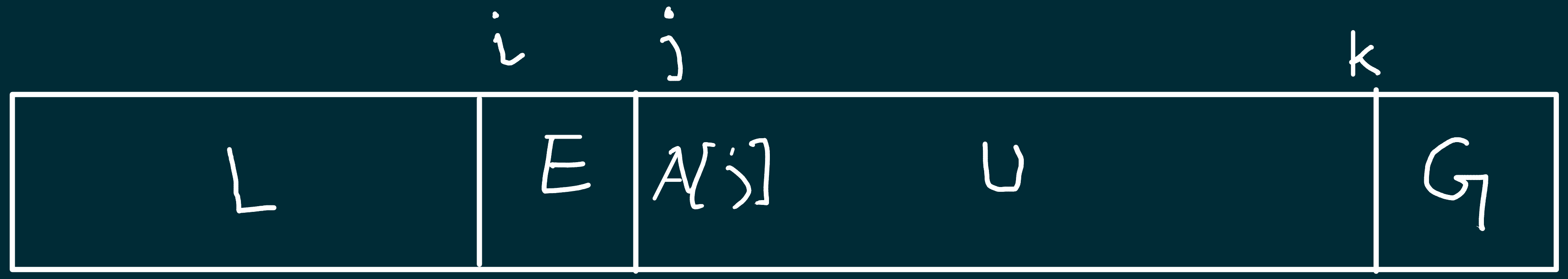
- Maintain only the values from two previous iteration

- v series need not be maintained

$$u_i a + v_i b = r_i$$

$$v_i = (r_i - u_i a)/b$$

Partitioning in quick sort



Choose $p = A[0]$ as pivot

- L — array elements $< p$
- E — array elements $= p$
- G — array elements $> p$
- U — unprocessed

$i = 0$
$j = 1$
$k = n-1$

```
i = 0; j = 1; k = n-1; p = A[0];
while (j <= k) {
    if (A[j] == p)    ++j
    else if (A[j] < p)
            swap A[i] with A[j]
             ++i; ++j;

     else
            swap A[j] and A[k]
             --k;
}
```

After n-1 iterations,

U becomes empty

Array is LEUG

LEG

<u>Exercise 1</u>: A is sorted in ascending order.

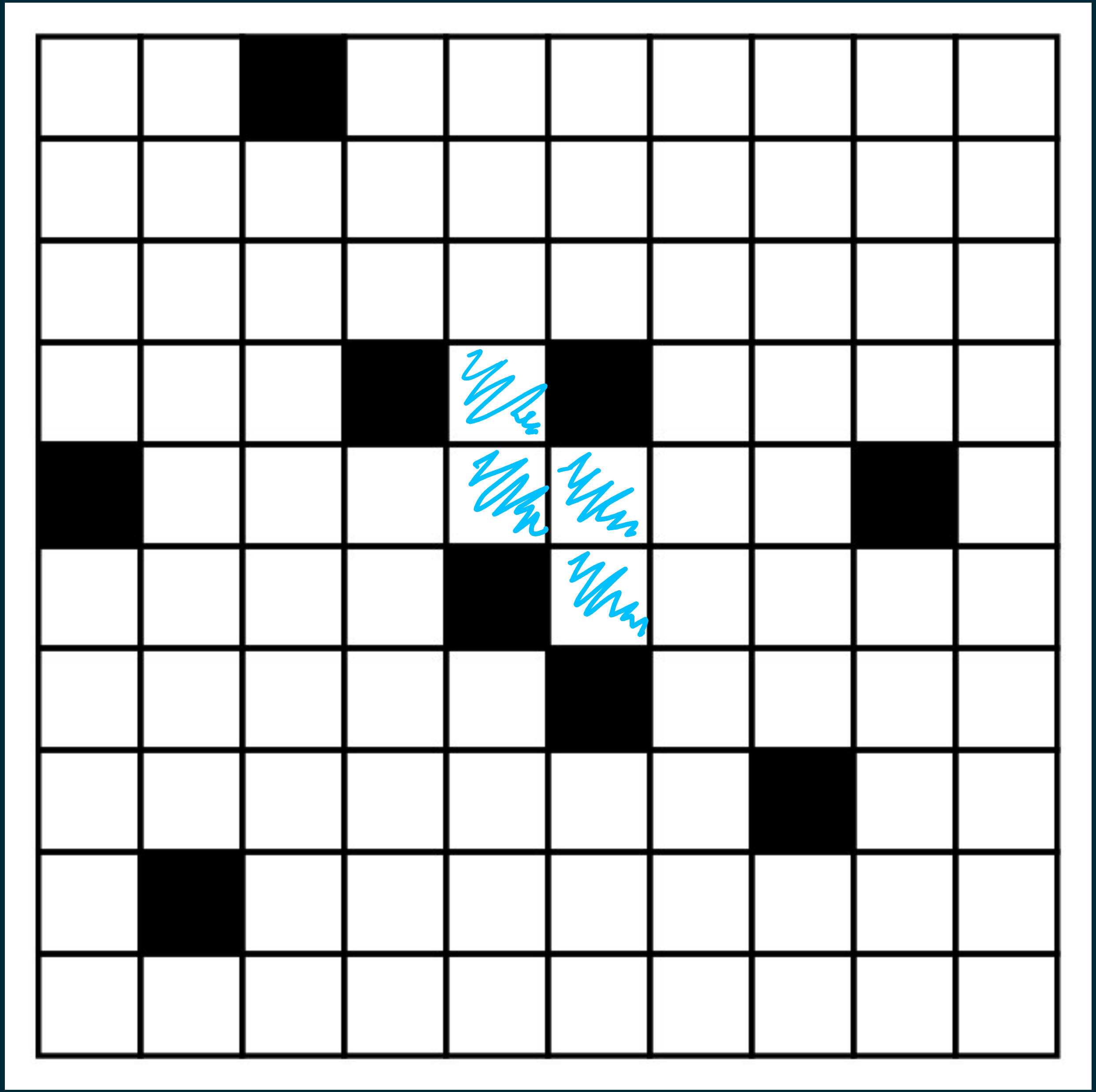
$O(n^{3/2})$ time

A is sorted in descending order

$O(n^2)$ time.

Exercise 2

If A contains r distinct values, then this quick sort runs in $O(rn)$ time.



(or more)

10x10 blocks.

9 blocks with COVID infection.

A block gets infection if two adjacent blocks are infected

Place the initial 9 infected cells in such a manner that the entire 10x10 grid is infected.

Or prove that it is impossible.