# **Sliding Window Substring Problems**

## **Identifying the Problem Constraint**

We need to identify what will be the main constraint will be to grow and shrink the window.

## **The Window**

* In the sliding window technique, we maintain a window.
* We grow or shrink the window depending on if we meet certain conditions (usually the main problem constraint).
* That window can be in 3 different states:
  1. **State 1: Growing**

While the window does not satisfy the problem constraint, we continue to grow the window until it does, updating any relevant local variables along the way.

* 1. **State 2: Satisfied the Constraint**

Once the problem constraint is satisfied, we update the global variable that is the required part of the problem requirement.

* 1. **State 3: Shrinking**

We now shrink the window UNTIL the constraint is no longer satisfied.

* 1. **Repeat:**

Now that the window constraint is no longer satisfied, we are back to State 1 and repeat this process until reaching the end of the collection

* The window is unstable if it violates the problem constraints, and it tries to stabilize by increasing or decreasing its size.

## **Global and Local Tracker Variables**

* Most sliding window techniques require you to keep global and local tracker variables.
  + **‘Find the minimum value’** 
    - keep a **min** value **global** tracker variable
    - keep a **min** value **local** value tracker variable to compare against the global
  + **‘Find the maximum value’** 
    - keep a **max** value **global** tracker variable
    - keep a **max** value **local** value tracker variable to compare against the global
  + **‘Find the longest subset/substring’** 
    - keep two global pointers to the front and end of subset/string. take the substring of the difference between the two pointers after the algorithm has finished executing.
  + **‘Find the shortest subset/substring’** 
    - keep two global pointers to the front and end of subset/string. take the substring of the difference between the two pointers after the algorithm has finished executing.
  + Contains value(s)

## **Algorithm Flow**

1. Use two pointers: start and end to represent a window.
2. Move end to find a valid window.
3. When a valid window is found, move start to try and shrink the smallest window possible find a smaller window.