POLYGON CLIPPING

- Each edge of the polygon must be tested against each **edge of the clipping** window, usually a rectangle.
- As a result, **new edges may be added**, and **existing edges** may be discarded, retained, or divided.
- Multiple polygons may result from clipping a single polygon.

SUTHERLAND HODGEMAN POLYGON CLIPPING

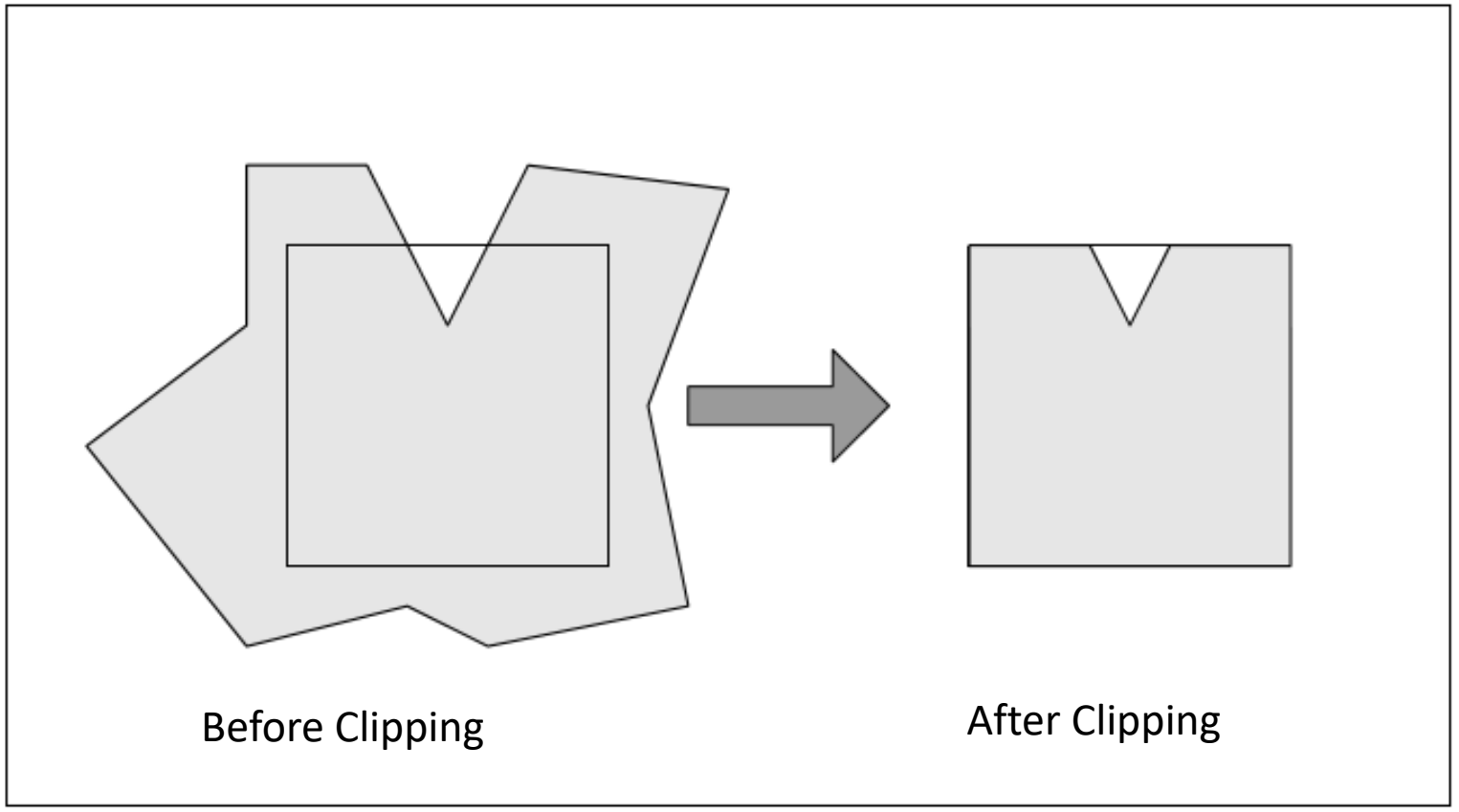
- This algorithm is based on a **divide and-conquer strategy** that solves a series of simple and identical problems that, when combined, solve the overall problem.

- The simple problem is to clip a polygon against a **single infinite clipping edge.**

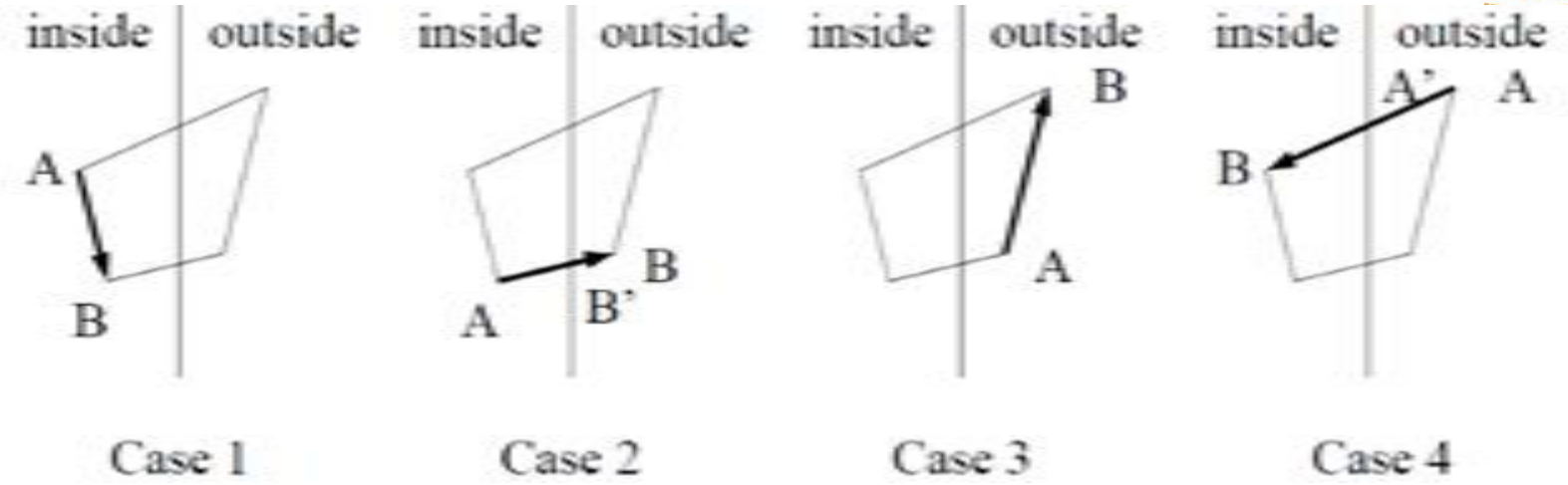
- This process outputs the **series of vertices** that define the clipped polygon

- **Four clipping** edges, each defining one boundary of the clipping window, are used to successively to fully clip the polygon.

SUTHERLAND HODGEMAN POLYGON CLIPPING



SUTHERLAND HODGEMAN POLYGON CLIPPING



CASE1: Inside→Inside---------End
CASE2: Inside→Outside-----Intersection
CASE3: Outside→Outside----Nothing
CASE4: Outside→Inside----- Intersection,End

SUTHERLAND HODGEMAN POLYGON CLIPPING CASES

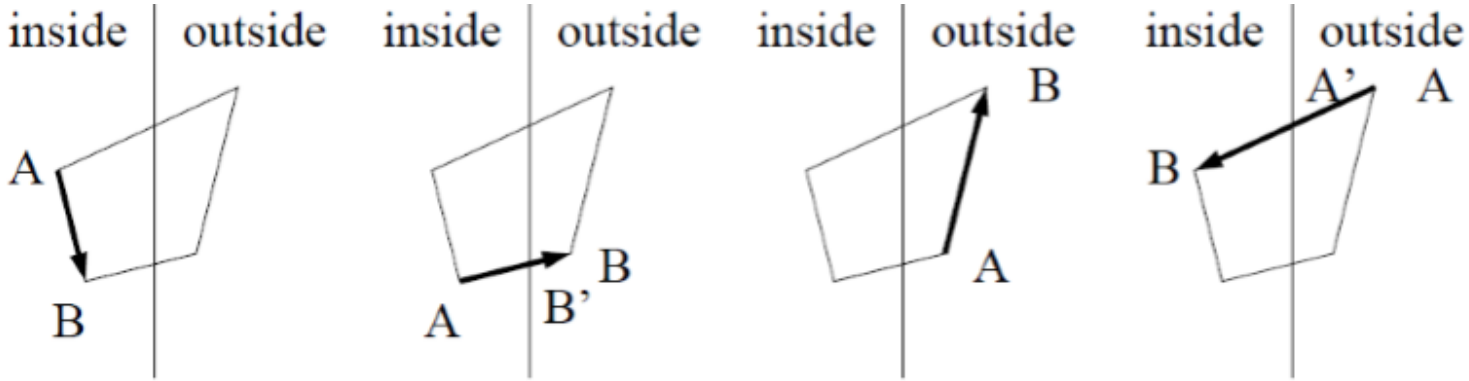
CASE1: Both vertices are inside : Only the second vertex is added to the output list

CASE2: First vertex is inside while second one is outside : Only the point of intersection of the edge with the clip boundary is added to the output list

CASE3: Both vertices are outside : No vertices are added to the output list

CASE4: First vertex is outside while second one is inside : Both the point of intersection of the edge with the clip boundary and the second vertex are added to the output list

SUTHERLAND HODGEMAN POLYGON CLIPPING



Case 1 Case 2 Case 3 Case 4

Assuming vertex A has already been processed,

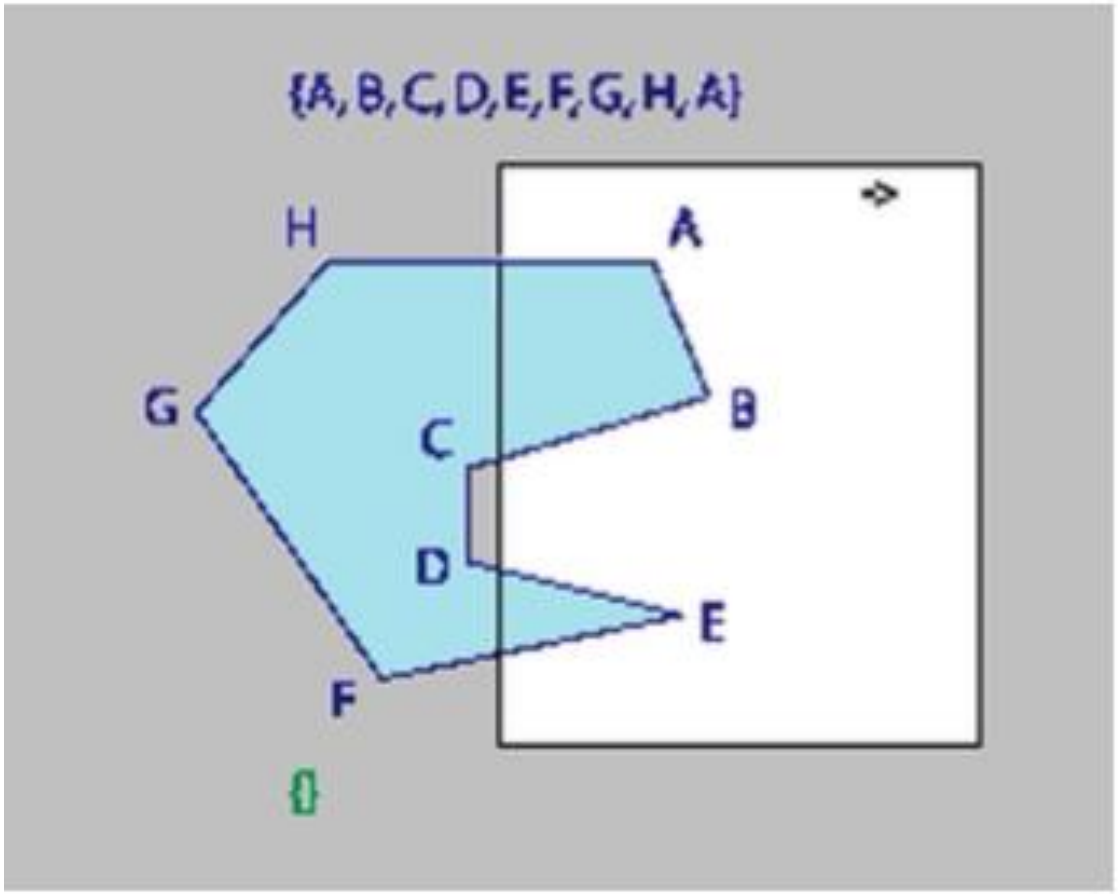
Case 1 — vertex B is added to the output list

Case 2 — vertex B' is added to the output (edge AB is clipped to AB')

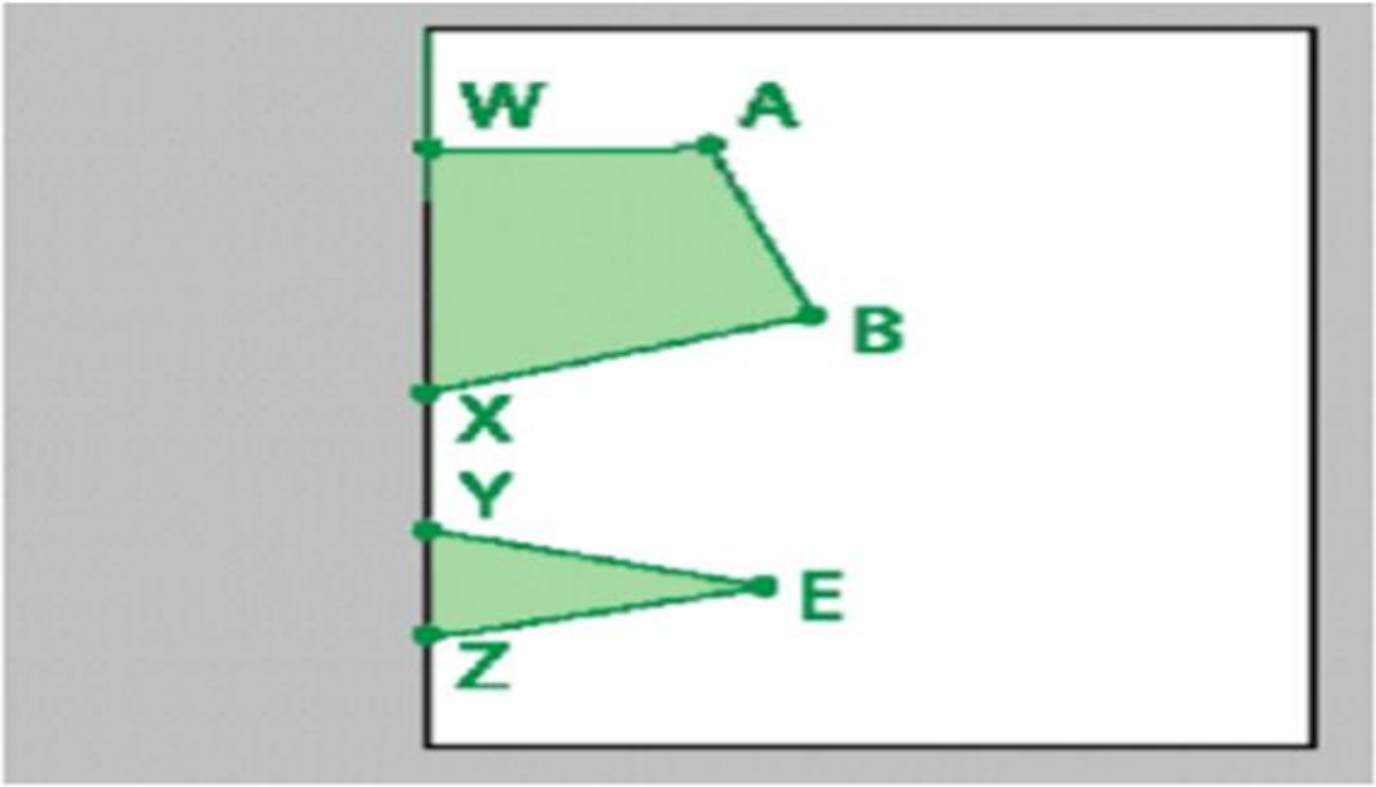
Case 3 — no vertex added (segment AB clipped out)

Case 4 — vertices A' and B are added to the output

SUTHERLAND HODGEMAN POLYGON CLIPPING

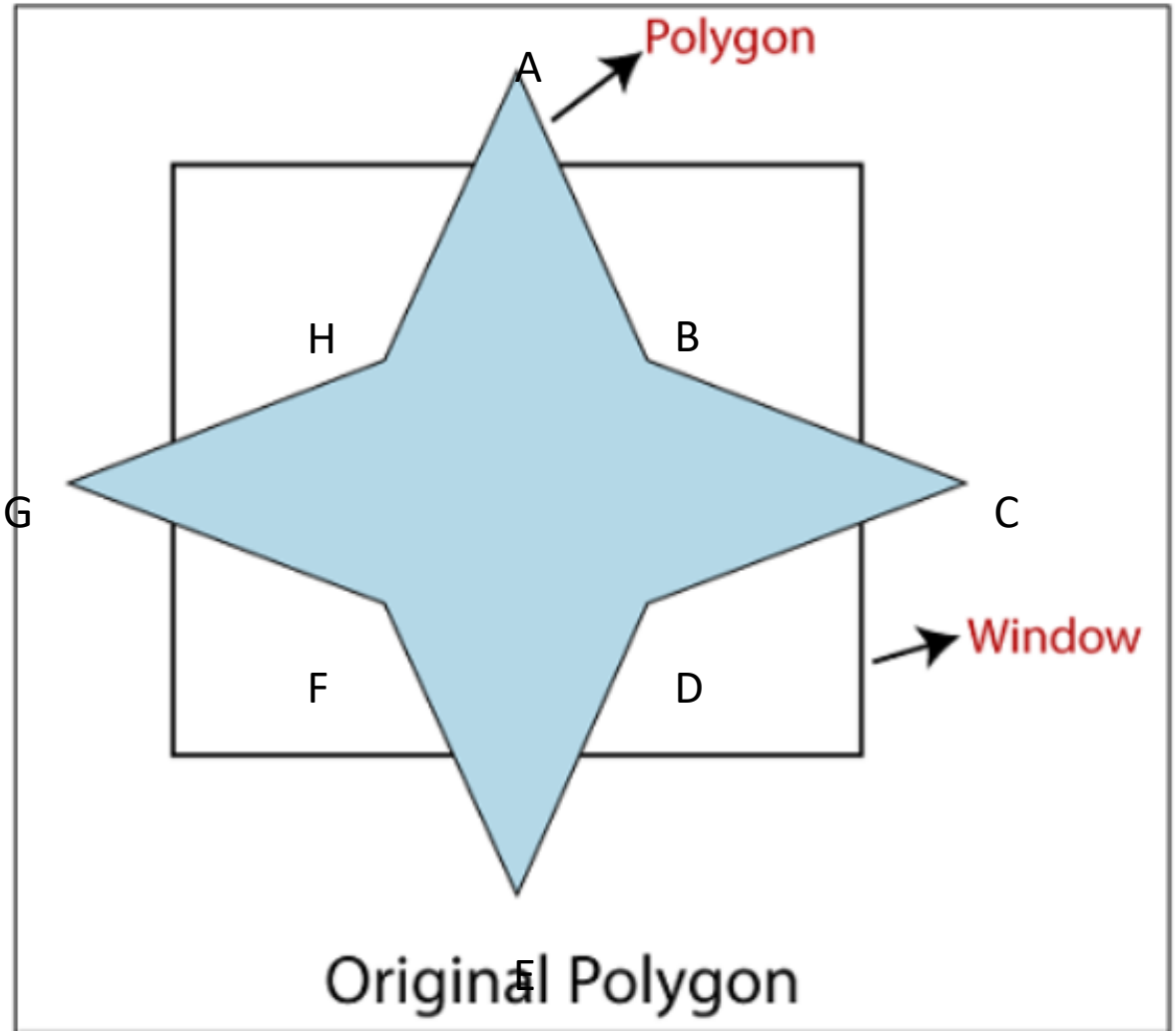


After Clipping



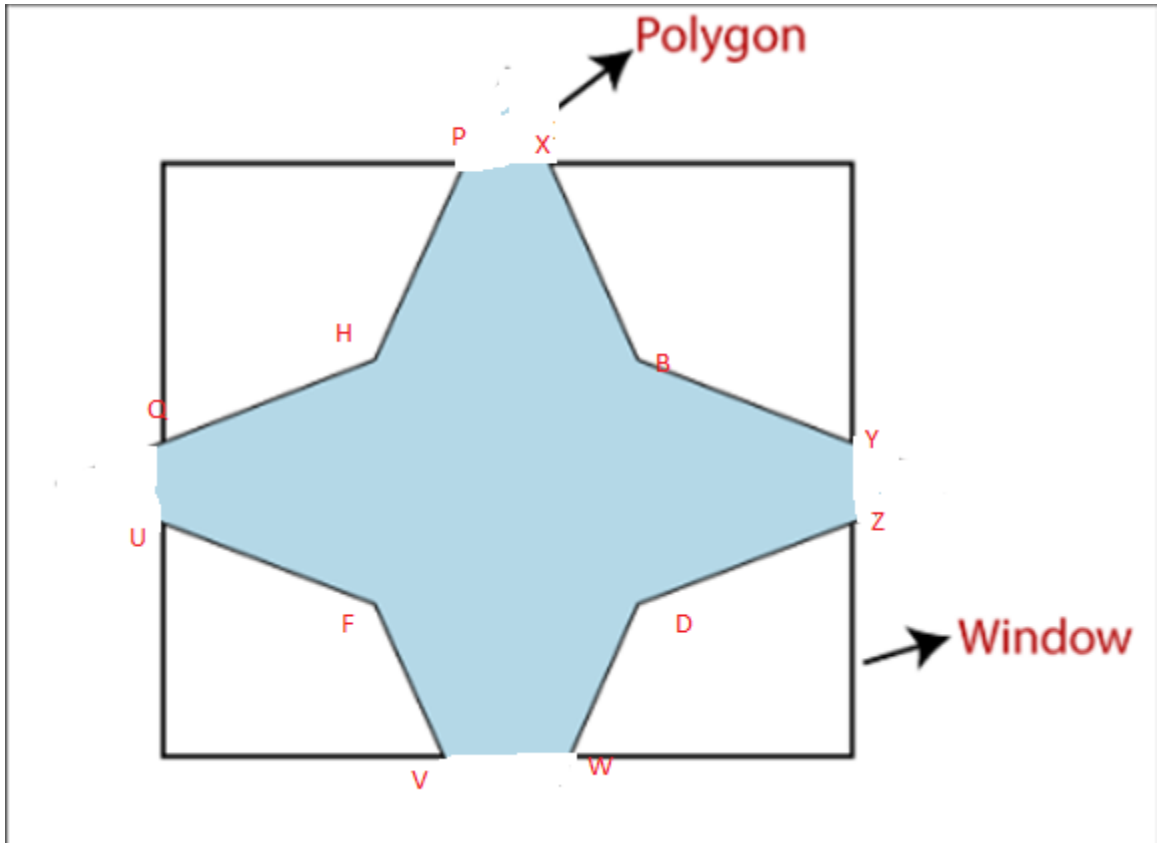
OUTPUT LIST→(B,X,Y,E,Z,W,A)

Home work



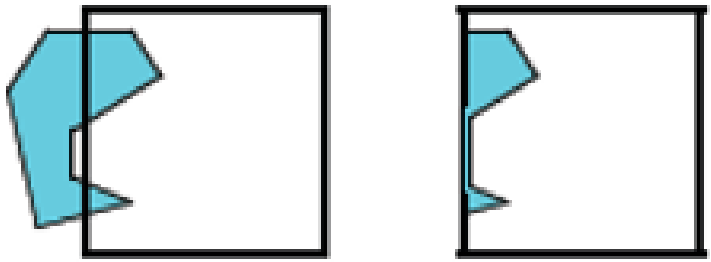
AFTER CLIPPING:

OUTPUT LIST→(X,B,Y,Z,D,W,V,F,U,Q,H,P)



Drawback of Sutherland Hodgeman Algorithm:

Clipping of the concave polygon → Can produce two CONNECTED areas



LOGICAL CLASSIFICATION OF INPUT DEVICES

- Locator Devices
- Stroke Devices
- Valuator Devices
- Choice Devices
- Pick Devices

Locator Devices

- To **select a co-ordinate** on the screen.
  - When the cursor is at the **desired position** on the screen, **a button** is clicked to select that co-ordinate point.
- **Mouse, joystick, trackballs, space balls, digitizers**.. Can be used as locator devices.




Stroke Devices

- A **sequence of co-ordinate points** can be selected.
- Locator devices in **continuous mode.**
- **Graphical tablet or digitizer** can also be used.

Valuator Devices:

- To give **scalar input values** like temperature and voltage levels.
- Floating point numbers within any range can be given as input
- E.g.: **Set of Control dials**.
- Rotation in one direction increases the values..
- **Slide-potentiometers, keyboards with a set of numeric keys can also be used**.
- **Display sliders, buttons, rotating scales and menus on screen**.

Choice Devices:

- To **construct a picture**.
- Selection from a list or menu of alternatives.
- Either a **keyboard or stand-alone "button box"** can be used for selecting item from the menu.
- The **selected screen position(x,y)** is compared with the menu option.
- **Touch panels** are also used.

Pick Devices:

- To **select parts of the screen** that are to be edited.
- **Choice devices** can be used here.
- If the selected point lies in the bounding rectangle of a particular object, then that'll be selected.
- If it lies in the area of 2 objects, then the squared distance from the point to the line segment both the objects are compared.

The different modes are:

- **Request mode**
- **Sample mode**
- **Event mode**

Some terminology:

- **measure** - information returned to user program
  - one or more characters from a keyboard
  - position for a locator
- **trigger** - manner in which device user can signal the computer that input is available
  - Enter key
  - button on locator

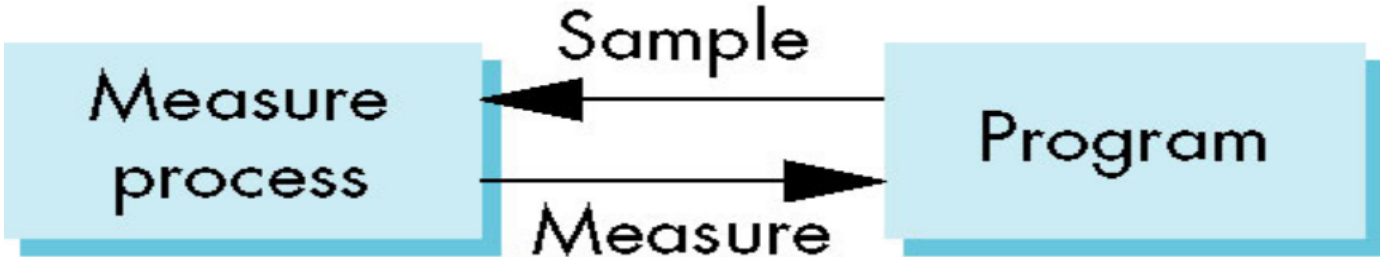
Request mode

- The measure of the device is not returned to the program until the **device is triggered.**


```
// Two keyboards, cin and cin2-- want to read from either
cin >> value;
cin2 >> value;     // Can't read until cin read returns
```

Sample mode

- Input is immediate - Measure is returned **immediately** after the function is called in the user program (device sample).
- **No trigger** needed
- Useful in apps where the program guides the user.

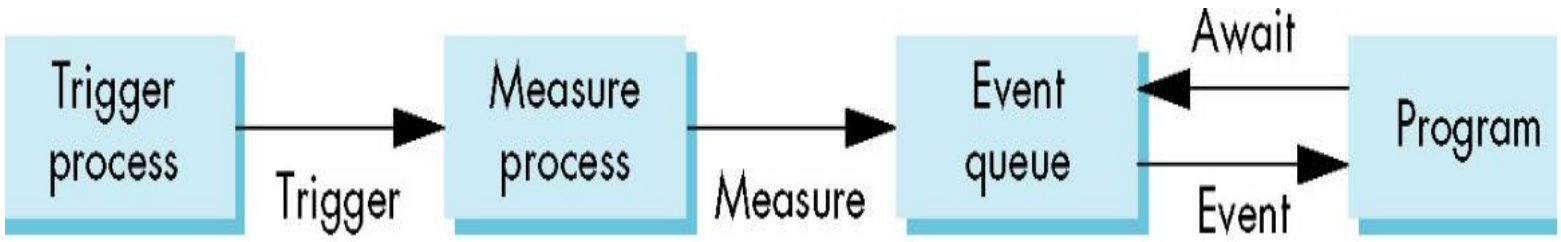


Sample mode - Example


```
while (!done) {
   if (cin.hasInput()) {
            cin >> value;
            done = true;
          }
   else if (cin2.hasInput()) {
            cin2 >> value;
            done = true;
          }
}
```

Event mode

- Measure is place on an event queue
- Can handle **multiple inputs**
- When device **triggered** an event is generated
- Identifier for device placed in the event queue
- Event queue process is independent of the application, asynchronous



Event mode

- Event queue processed by application

```
while (true) {
                if (!eventQ.isEmpty()) {
                        event = eventQ.remove();
                        // Handle the event
                        if (event.source == cin)
                                process cin input
                        else
                                process cin2 input
                }
}
```

Event mode

- Event queue processed by system which invokes **callback functions**

```
void display() {
                    ...
}

glutDisplayFunc(display); // register event handler
glutMainLoop();   // allow the system to take over and wait for events
```