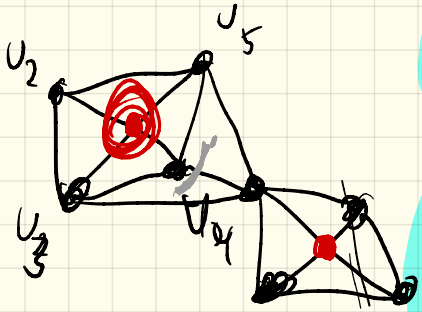
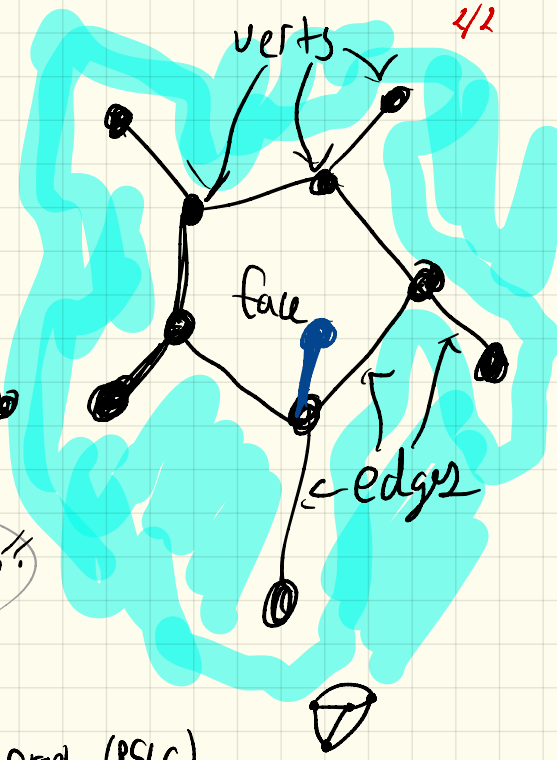
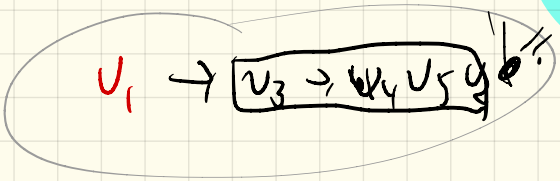
DCEL



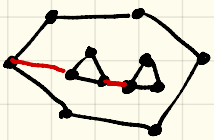




How do we rep a planar straight line graph (PSLG)

Edge based rep (DCEL)

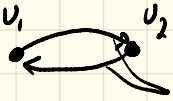
3 record types
- vertex records
- edge records
- face records



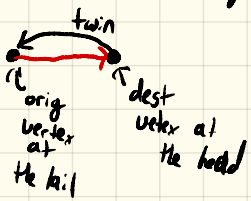
- dummy edge
- real edges

- Assume that all faces have no holes
  ⇒ edges bounding a face form a cyclic list

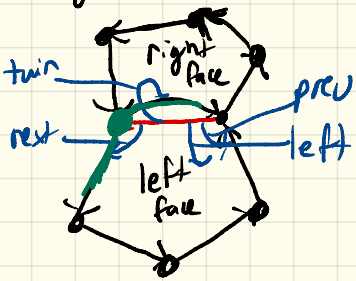
Edge: (u_1, u_2) 



2 directed edges or "half edges"



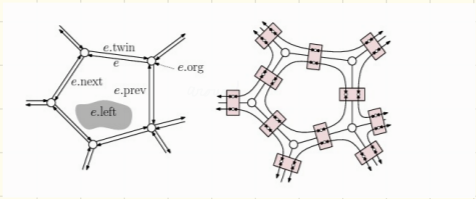
- head and tail of half edge
- twin of edge
- left and right face



Store;
- orig: pointer to orig vertex
- twin: pointer to oppisitely directed edge
- left: pointer to incident face (left)
- next: pointer to next half-edge ccw around incident face
- prev: pointer to previous half edge ccw around incident face

Vert:
- coord: vertex coord
- inc_edge: pointer to any incend directed edge by this vetex as orig

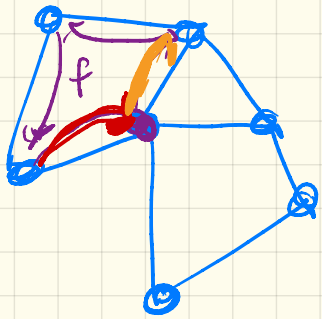
Face;
- inc_edge: pointer to any incedent edge

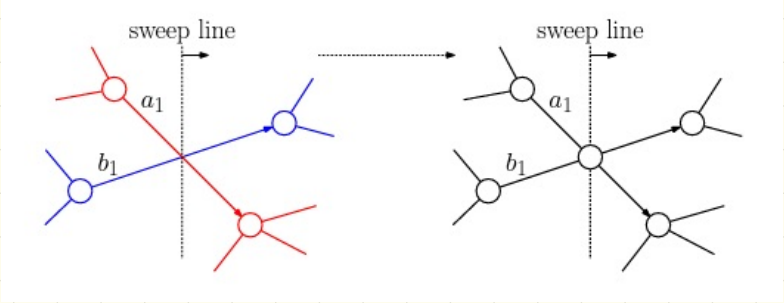


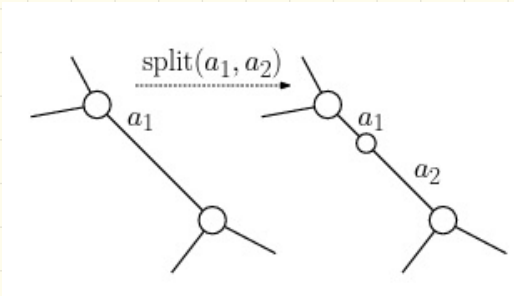
Write algo to
enumerate all verts of a face

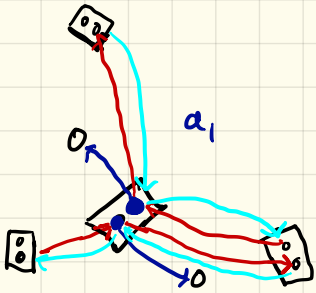
enumerate_vertices(Face f)

```
enumerate_vertices(Face f) {
    Edge start = f.inc_edge;
    Edge e = start;
    do {
        output e.org;
        e = e.next;
    } while (e != start);
}
```



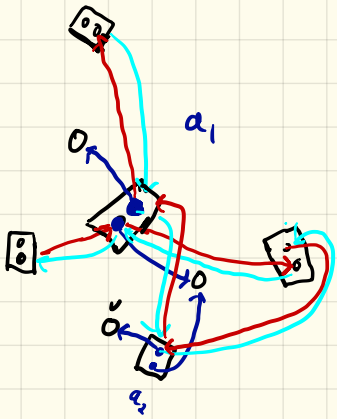


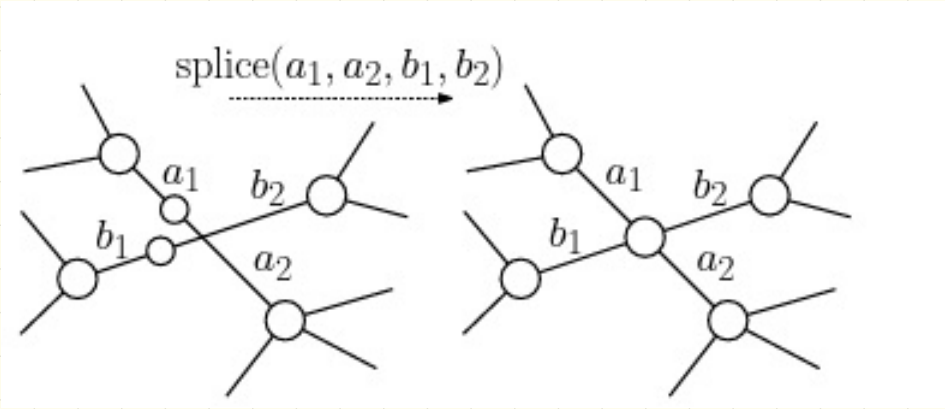




Split an

```
split(edge a1, edge a2) {                  // a2 is returned
    a2 = new edge(v, a1.dest());           // create edge (v,a1.dest)
    a2.next  = a1.next;    a1.next.prev  = a2;
    a1.next  = a2;         a2.prev  = a1;
    a1t = a1.twin;         a2t = a2.twin; // the twins
    a2t.prev = a1t.prev;   a1t.prev.next = a2t;
    a1t.prev = a2t;        a2t.next  = a1t;
}
```




splice(a_1, a_2, b_1, b_2)
a_1
b_2
b_1
a_2
a_1
b_2
b_1
a_2