of vertices and edges. Since each edge defines its vertices explicitly, a connected sequence of edges describes a face sufficient. If this sequence is circular the face is called bounded. *Note:* This definition does not depend on what kind of geometric structure a DCEL represents, a (connected) planar graph in 2D or a polyhedron in 3D.

Example 1. The following example depicted in Figure 3 shows a simple polygon with a dangling edge represented as a DCEL, where A is the outer face an B is the interior one. Note: the dangling edge a has two equal adjacent faces, i.e. ($F1 = F2 = A$).

E	V1	V2	F1	F2	P1	P2
a	1	2	A	A	a	c
b	2	5	A	B	a	e
c	2	3	B	A	b	d
d	3	4	B	A	c	e
e	4	5	B	A	d	b

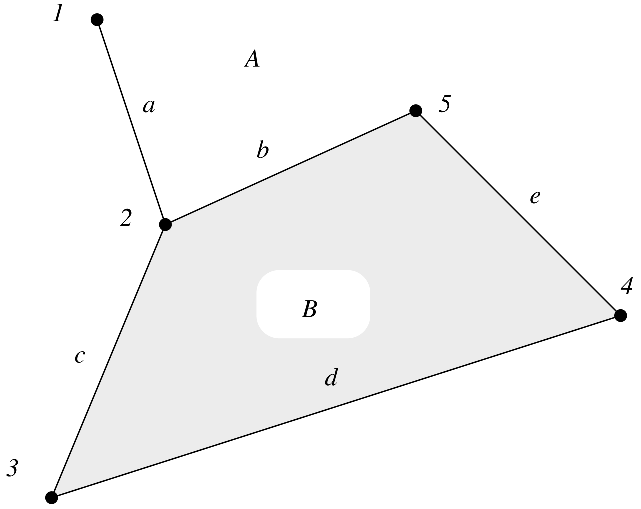

Figure 3: Example 1 with dangling edge.

Example 2. In Figure 4 is shown a more practical but not too complicated example with three faces and without dangling elements.

E	V1	V2	F1	F2	P1	P2
a	1	5	A	B	e	b
b	1	2	B	A	a	f
c	2	3	B	C	b	d
d	3	4	B	C	c	e
e	4	5	B	C	d	f
f	2	5	C	A	c	a

Here we give some important type definitions, which will be used in some of the programs. I is an interface or traits class. It holds a lot of special technical definitions and types necessary for an actual implementation of the given programs (for a detailed information of I see the appendix).
For storing the elements of a DCEL we use the following STL-containers: