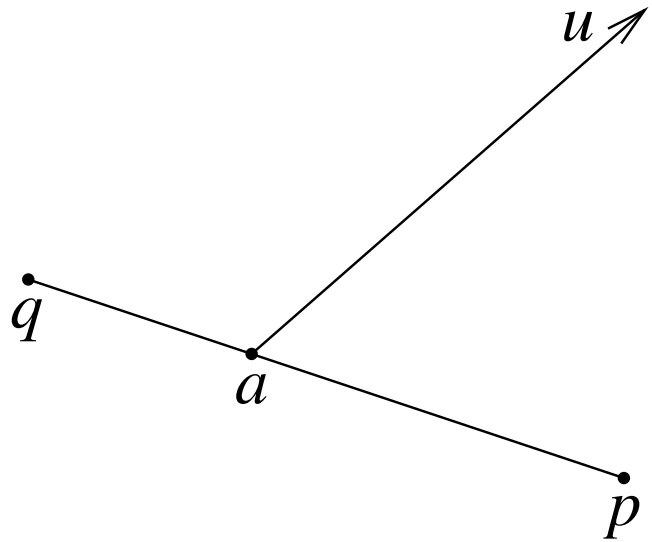
Homework 2 Solution

Elisha Sacks

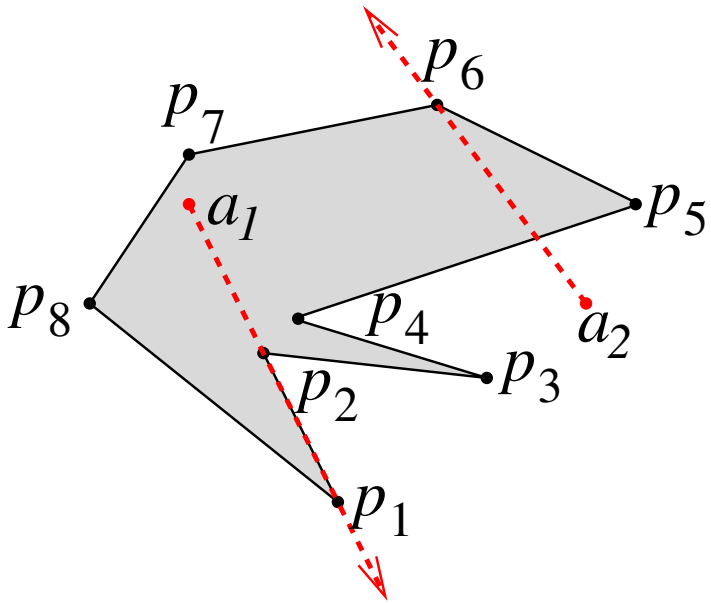
Problem 1



What are the degenerate inputs?

The point a is a vertex p or is on an edge pq of the polygon P.

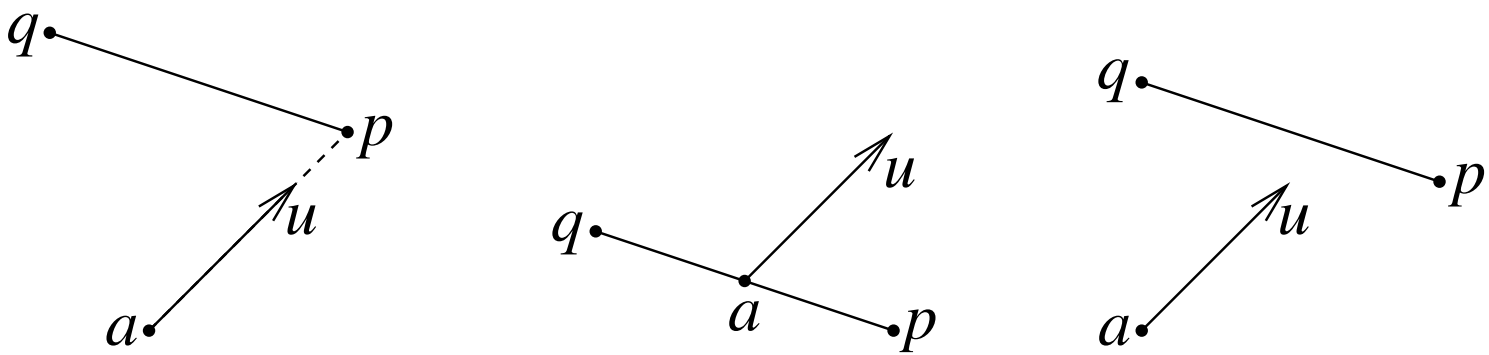
Problem 2



When is a predicate degenerate on a nondegenerate input?

The ray $a + ku$ contains a vertex or an edge of P.

Problem 3



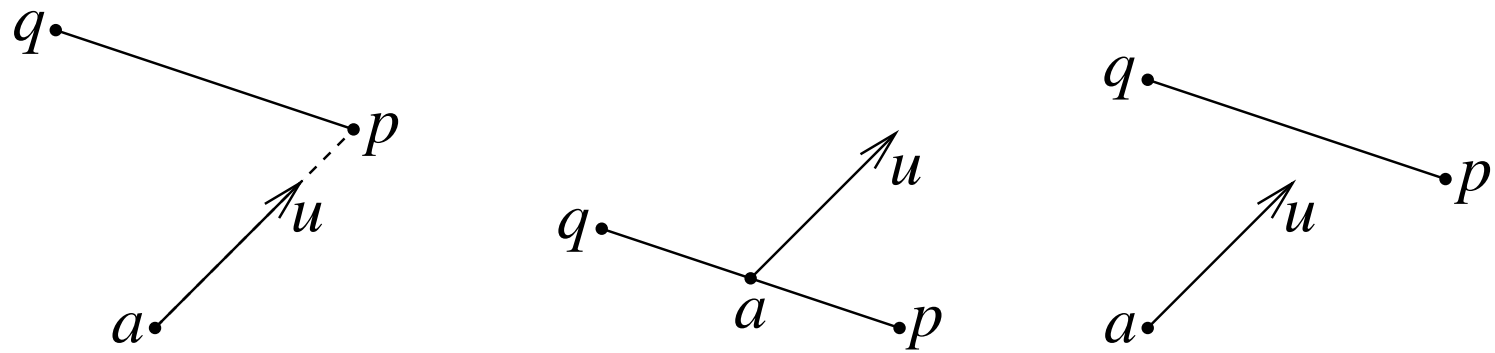
Specify the algorithm in English.

1. set $r = -1$ and set u to a random vector
2. for each edge pq of P
    3. if $a + ku$ intersects $[p, q)$
        3.1 if a is collinear with pq return 0
        3.2 if u points into the opposite side of pq from a
            set $r = -r$
4. return r

Problem 4: ACP implementation

```
int pointInPoly (Point *a, const Points &pts)
{
  PTR<Point> u = new Point(1, 1);
  int r = -1;
  unsigned int n = pts.size();
  for (unsigned int i = 0u; i < n; ++i) {
    unsigned int j = (i + 1u)%n;
    int s = rayEdgeIntersection(a, u,
                                pts[i], pts[j]);
    if (s == 0)
      return 0;
    else if (s == 1)
      r = - r;
  }
  return r;
}
```

Problem 4 (continued)




```
int rayEdgeIntersection (Point *a, Point *u,
                         Point *p, Point *q)
{
  int s1 = PointRaySide(p, a, u),
      s2 = PointRaySide(q, a, u);
  if (s1*s2 == 1 || s1 != 0)
    return -1;
  int s3 = LeftTurn(p, q, a),
      s4 = PointRaySide(p, q, u);
  return - s3*s4;
}
```

Problem 4 (continued)

```
class PointRaySide : public Primitive {
  Point *p, *a, *u;

  DeclareSign {
    PV2<N> pp = p->get<N>(), aa = a->get<N>(),
           uu = u->get<N>();
    return uu.cross(pp - aa);
  }
public:
PointRaySide (Point *p, Point *a, Point *u)
  : p(p), a(a), u(u) {}
};
```