In article <jonas-y.734802983@gouraud> jonas-y@isy.liu.se (Jonas Yngvesson)
writes:

> Intersection Between a Line and a Polygon (UNDECIDABLE??),
> by Dave Baraff, Tom Duff
> From: deb@charisma.graphics.cornell.edu
> In recent years, many geometric problems have been successfully modeled in a
> new language called PostScript. (See "PostScript Language", by Adobe Systems
> Incorporated, ISBN # 0-201-10179-3, co. 1985).
> So, given a line L and a polygon P, we can write a PostScript program that
> draws the line L and the polygon P, and then "outputs" the answer. By
> "output", we mean the program executes a command called "showpage", which
> actually prints a page of paper containing the line and the polygon. A quick
> examination of the paper provides an answer to the reduced problem Q, and
thus
> the original problem.

Curiously, in modern PostScript, the point in a polygon problem can
be solved even more easily. To wit:

```
%%Title: Point in Polygon
%%Creator: Allen B (ab@cc.purdue.edu)
%%For: the amusement of comp.graphics regulars
%%LanguageLevel: 2
%%DocumentNeededResource: humor sense thereof
%%EndComments
% This program will test whether a point is inside a given polygon.
% Currently it uses the even-odd rule, but that can be changed by
% replacing ineofill with infill.  These are Level 2 operators,
```

```
% so if you've only got Level 1 you're out of luck.
% The result will be printed on the output stream.
% Caution: only accurate to device pixels!
% Put a huge scale in first if you aren't sure.
% Point to test
% Vertices of polygon in counter-clockwise order
dup 0 get aload pop moveto dup length 1 dup 3 1 roll
sub getinterval { aload pop lineto } forall closepath
ineofill { (Yes!) } { (No!) } ifelse =
```