- Anwendung der hierarchischen Darstellung:
  - Wir berechnen eine Annäherung der Lsg für P_0, da $P_0 \leq 4$ Ecken hat geht dies in $O(1)$ (4 ist eine Konstante)
  - Durchlaufe die Folge $P_0, P_1, \dots, P_k$ und verfeinere die Lsg schrittweise
  - Am Ende haben wir die Lsg für $P_k = P$

- Darstellung im Rechner:
  - P_i $(0 \leq i \leq k)$ als doppelt verkettete Liste der Ecken und jede Ecke von P_i zeigt zusätzlich auf ihre Kopie in P_{i+1} und umgekehrt.

```
1 - 2 - 3 - 4 - 5 - 6 - 7 - 8 - 9      P2 = P
    |       |       |       |
    2 ----- 4 ----- 6 ----- 8          P1
            |               |
            4 ------------- 8          P0
```

  - Explizite Speicherung des Kantenbaums
    ⇝ Dynamisierung (Ecken einfügen und löschen wie bei Suchbäumen)

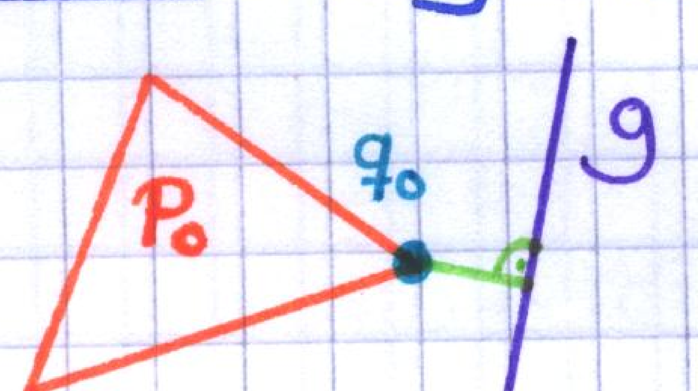
Welche Darstellung ist besser? Ist wahrscheinlich egal

2.2 Anwendung I: Schnitt Polygon / Gerade

Geg.: Hierarchische Darstellung $P_0, \dots, P_k$ eines konv Polygons P und eine Gerade g

Ges.: $P \cap g = \begin{cases} 2 \text{ Pkte} \\ 1 \text{ Pkt} \\ \emptyset \end{cases}$ wenn wir nur den Rand von P betrachten. Sonst $P \cap g = \begin{cases} \text{Strecke} \\ 1 \text{ Pkt} \\ \emptyset \end{cases}$

- Algorithmus: Starte mit $i=0$
  Betrachte $P_0 \cap g \Rightarrow$ 2 Fälle
  Fall 1: $P_0 \cap g = \emptyset$



  - Finde $q_0 \in P_0$ mit minimalem Abstand zu g (Minimumssuche auf e-elementiger Menge $e \leq 4$)
  - $i=0$
  - (1) if $i==k$ then (dh if $P_i == P$)
    Algorithmus ist fertig. Ausgabe: $P \cap g = \emptyset$ und $q_0 \in P_0$ liegt am nächsten zu g
    end if
  - Übergang $P_i \to P_{i+1}$
  - Beobachtung:
    Kandidaten für neuen Pkt q_{i+1} sind genau die maximal 7 Eckpkte von P_{i+1}, die aus der Verfeinerung der beiden Nachbarkanten e_1 und e_2 (e_1 und e_2 sind die Kanten aus P_i die sich im Pkt q_i treffen) entstehen.
    Dies folgt aus der Konvexität von P_{i+1}.
  - Teste, ob in dieser konst # von Kandidaten ein Pkt auf der anderen Seite von der Geraden g liegt (mit Orientation)