MAC 5747 – Geometria Computacional
Prof. José Coelho de Pina
Monitor: Fábio Henrique Viduani Martinez

LISTA DE EXERCÍCIOS 3 – RESPOSTAS

3. (Exercício 1.6.4.1 de [1] — *Integer overflow* Nos computadores onde variáveis do tipo inteiro podem assumir valores entre -2^{31} e $+2^{31}$, qual é o maior valor que podemos atribuir às coordenadas dos pontos a, b e c para que não ocorra *integer overflow* na função *Área2* (veja Primitivas)?

   Uma parcela de *Área2* deve ser $< 2^{31}$. Como são 6 as parcelas deste cálculo, cada uma delas deve ser $< 2^{28}$ e assim, cada coordenada deve ser $< 2^{14}$. Este é então o maior valor que podemos atribuir a uma coordenada dos pontos a, b e c para que não ocorra *integer overflow* na função *Área2*.

   Para uma resposta mais sofisticada e genial, veja

   `http://www.ime.usp.br/~coelho/geocomp2000/material/primitivas/arithmetic.ps`

   seção 15, página 8, onde uma rotina de propósito geral, chamada `sign-test`$(x_1, x_2, x_3, y_1, y_2, y_3)$, computa um inteiro de precisão simples que tem o mesmo sinal do produto

   $$x_1 * y_1 + x_2 * y_2 + x_3 * y_3$$

   quando temos $-2^{29} < x_1, x_2, x_3 < 2^{29}$ e $0 \leq y_1, y_2, y_3 < 2^{29}$.

6. (Exercício 1.6.4.2 de [1]) Descreva o que o predicado `IntersectProp` faz se o comando **if** é removido. Note que depois desta remoção o predicado `Intersect` ainda funciona.

   O predicado `IntersectProp` funcionará da mesma forma que o predicado `Intersect`, exceto quando ocorrerem intersecções impróprias.

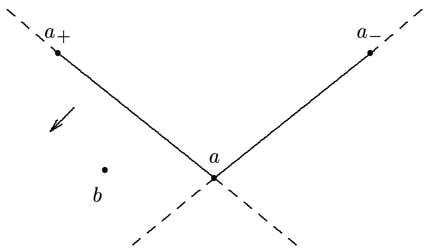
8. (Exercício 1.6.4.5 de [1] — `InCone`) Prove que ab está no cone no vértice a se e somente se no máximo uma das seguintes chamadas de `Left` devolve FALSE: `Left`(a, a_+, b), `Left`(a, b, a_-),`Left`(a, a_-, a_+), onde a_+ e a_- são os vértices sucessor e predecessor de a na fronteira de P, respectivamente.

   Podemos dividir esta demonstração em dois casos, a saber:

   Caso 1. `Left`(a, a_-, a_+) = TRUE.

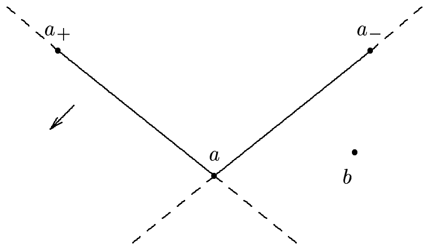
   Neste caso, temos três possibilidades:

   Subcaso 1.1



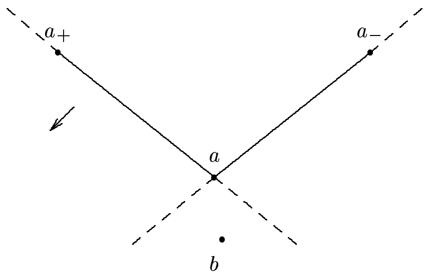
   Observe na figura que `Left`(a, a_+, b) = TRUE e `Left`(a, b, a_-) = FALSE.

Subcaso 1.2



Observe na figura que $\texttt{Left}(a, a_+, b) = $ FALSE e $\texttt{Left}(a, b, a_-) = $ TRUE.

Subcaso 1.3

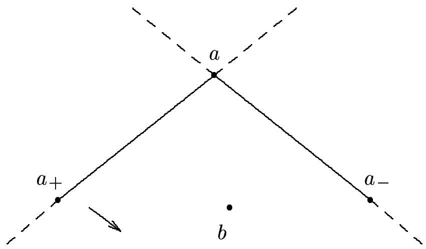


Observe finalmente que $\texttt{Left}(a, a_+, b) = $ TRUE e $\texttt{Left}(a, b, a_-) = $ TRUE.

Nestes três subcasos, temos que no máximo uma das chamadas $\texttt{Left}(a, a_+, b)$, $\texttt{Left}(a, b, a_-)$ e $\texttt{Left}(a, a_-, a_+)$ devolve FALSE.

Caso 2. $\texttt{Left}(a, a_-, a_+) = $ FALSE:

Temos apenas uma possibilidade:



É fácil ver que $\texttt{Left}(a, a_+, b) = $ TRUE $= \texttt{Left}(a, b, a_-)$. Novamente, temos que no máximo uma das chamadas $\texttt{Left}(a, a_+, b)$, $\texttt{Left}(a, b, a_-)$ e $\texttt{Left}(a, a_-, a_+)$ devolve FALSE.