Programming Assignment #9

CS 200, Fall 2017

Due Wednesday, November 29

Implement a class for clipping line segments and polygons to a convex polygon. I will supply you with the header file `Clip.h`, which has public and private declarations for this class:

```
class Clip {
  public:
    Clip(void);
    Clip(const std::vector<Point>& clip_obj_verts);
    bool operator()(Point& P, Point& Q);
    bool operator()(std::vector<Point>& verts);
  private:
    std::vector<HalfPlane> half_planes;
    std::vector<Point> temp_vertices;
    bool clip_to_halfplane(const HalfPlane& h, std::vector<Point>& verts);
};
```

(the files `Affine.h` and `HalfPlane.h`, as well as the standard header file `vector` have been included). The member functions are described below. Your are free to use the private data members as you see fit. However, the private function `clip_to_halfplane` must be implemented as described below.

`Clip()` — (default constructor) creates an instance of the class for clipping to the standard square; i.e., the clipping object is the square with vertices $(\pm 1, \pm 1)$.

`Clip(clip_obj_verts)` — (non–default constructor) creates an instance of the class for clipping to the convex polygon with vertices stored in the vector `clip_obj_verts`. The vertices should be assumed to be those of a convex polygon, ordered either in a clockwise or counterclockwise manner.

`operator(P,Q)` — computes the result of clipping the line segment $\overline{PQ}$ to the clipping object. If there is no intersection between the line segment and the clipping object, the return value is `false`. The values of P and Q are not modified in this case. Otherwise, the return value is `true`, and the values of P and Q are modified to hold the endpoints of the clipped line segment.

`operator()(verts)` — computes the result of clipping the polygon $\mathcal{P}$ with vertices stored in the vector `verts` to the clipping object. If there is no intersection between $\mathcal{P}$ and the clipping object, the return value is `false`. Otherwise, the return value is `true`, and the vector `verts` is modified to hold the vertices of the clipped polygon. Note that `verts` may be modified as a result of calling this function, even if the return value is `false`.

`clip_to_halfplane(h,verts)` — computes the result of clipping the polygon $\mathcal{P}$ with vertices stored in the vector `verts` to the half–plane h. If there is no intersection between $\mathcal{P}$ and h, the return value is `false`. Otherwise, the return value is `true`, and the vector `verts` is modified to hold the vertices of the clipped polygon. Note that this function does *not* use the clipping object. However, you may wish to use the private datum `temp_vertices` to store any temporary vertices created in the clipping process.

For the line clipping function, you are expected to use the parametric line clipping algorithm discussed in class (Liang–Barsky algorithm). And for the polygon clipping function, you are to use the Sutherland–Hodgman polygon clipping algorithm. Part of your grade will be for the efficiency of your implementations.

Your submission should consists of a single file: the implementation file `Clip.cpp`. You may only include the header files `Affine.h`, `HalfPlane.h`, `Clip.h`, and the standard C++ header file `vector`.