실습과제 09

1. 선분 절단 알고리즘 구현하기
   - 코헨-서덜랜드의 알고리즘 구현: clipLine_CS()
   - 리앙-바스키 알고리즘 구현: clipLine_LB()
   - 테스트 코드 작성
     - 다음 코드 참조
     - 콘솔 응용 프로그램으로 작성해도 됨
     - 여러 영역의 점들을 이용해 테스트할 것

16

```
void lineClipTest(Point2D min, Point2D max, Point2D p, Point2D q) {
    printf("\n원선분: (%4.2f,%4.2f),(%4.2f,%4.2f)\n", p.x, p.y, q.x, q.y);
    printf("방법 1: ");
    clipLine(min, max, p, q);
    printf("방법 2: ");
    clipLine2(min, max, p, q);
}

int main(int argc, char ** argv)
{
    Point2D min = { -1,-1 };
    Point2D max = {  1, 1 };
    Point2D p[8] = {{ -2, 0 }, { 2, 0 },
                    { -0.3, 0 },{ 2, 1.5 },
                    { -2, 0 },{ 0.5, -0.2},
                    { -1.3, 0 },{ -2, 1.5 } 
    };

    lineClipTest(min, max, p[0], p[1]);
    lineClipTest(min, max, p[2], p[3]);
    lineClipTest(min, max, p[4], p[5]);
    lineClipTest(min, max, p[6], p[7]);

    getchar();
    return 0;
}
```

```
D:₩저서-그래픽스₩교재코드₩08장-절단과가시성판단₩Debug₩LineClipping.exe

원선분: (-2.00,0.00),(2.00,0.00)
방법 1:          절단후: (1.00,0.00)--(-1.00,0.00)
방법 2:          절단후: (-1.00,0.00)--(1.00,0.00)

원선분: (-0.30,0.00),(2.00,1.50)
방법 1:          절단후: (1.00,0.85)--(-0.30,0.00)
방법 2:          절단후: (-0.30,0.00)--(1.00,0.85)

원선분: (-2.00,0.00),(0.50,-0.20)
방법 1:          절단후: (-1.00,-0.08)--(0.50,-0.20)
방법 2:          절단후: (-1.00,-0.08)--(0.50,-0.20)

원선분: (-1.30,0.00),(-2.00,1.50)
방법 1: 방법 2:
```

17