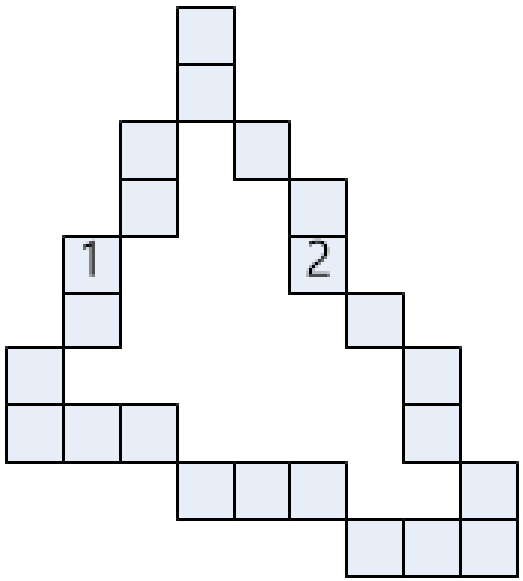
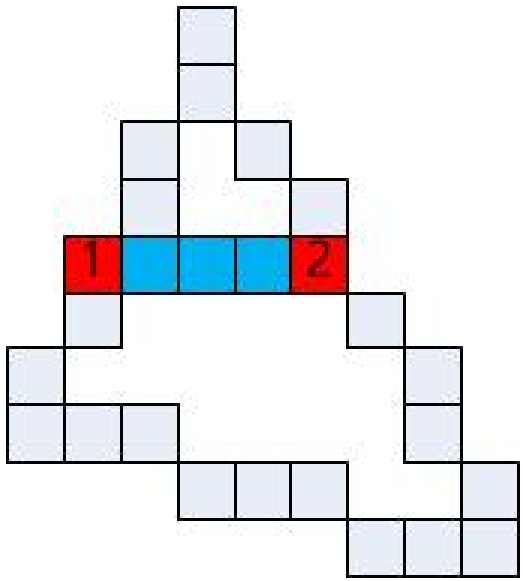
Draw Filled Triangle

C++ Equality and Equivalence

For the types that are both EqualityComparable and LessThanComparable, the C++ standard library makes a distinction between **equality**, which is the value of the expression a == b and **equivalence**, which is the value of the expression !(a < b) && !(b < a).

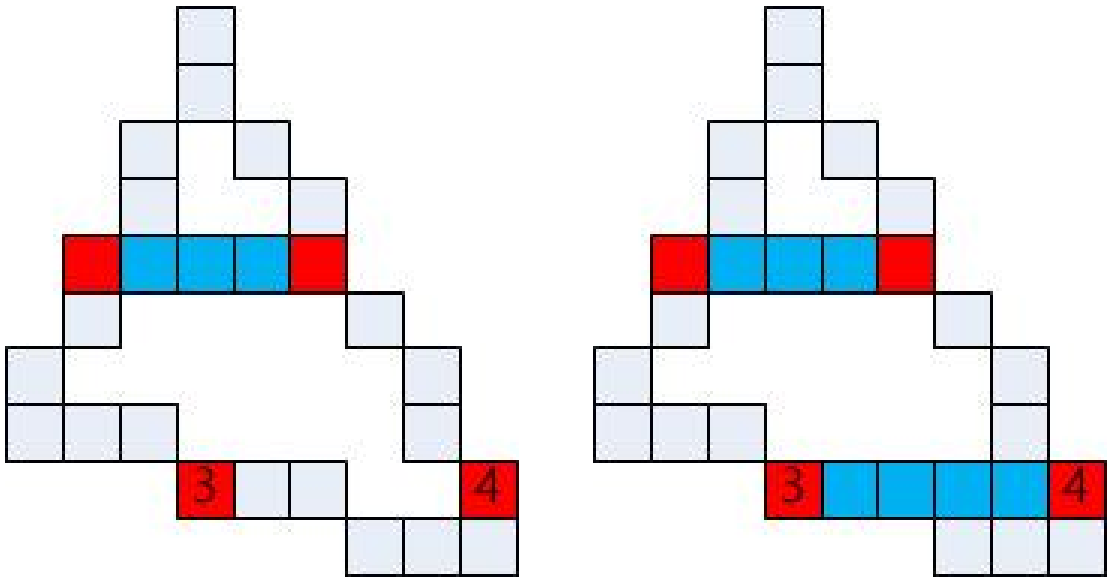
Draw Filled Triangle





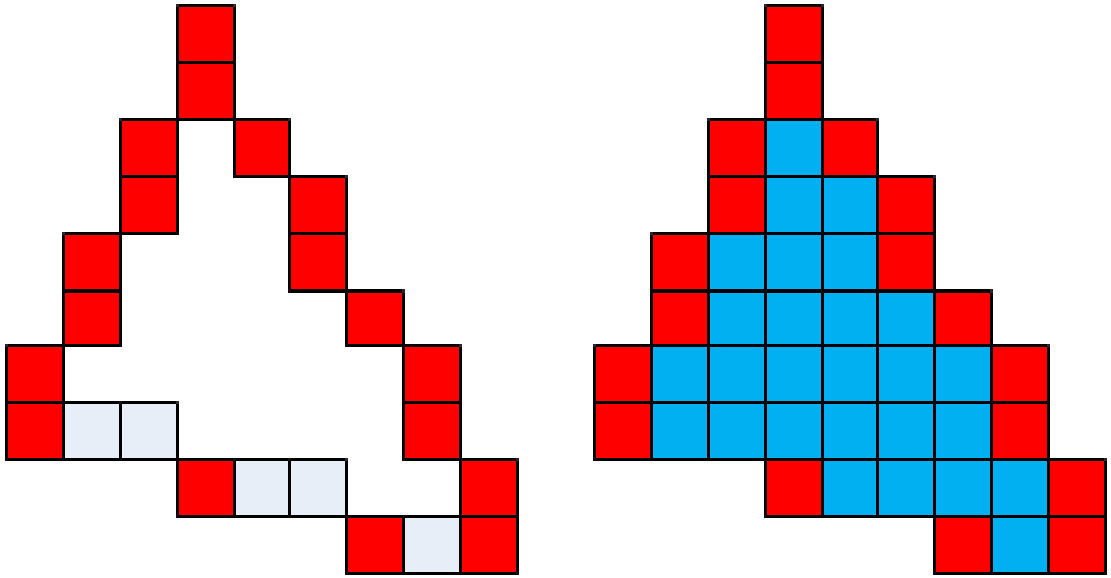
[Fig] pixel 1 is begin point, pixel 2 is end point

[Fig] Draw horizontal line between 1 and 2



[Fig] More than 2 pixel can be exist, so select left most and right most one.

[Fig] Draw horizontal line from 3 to 4



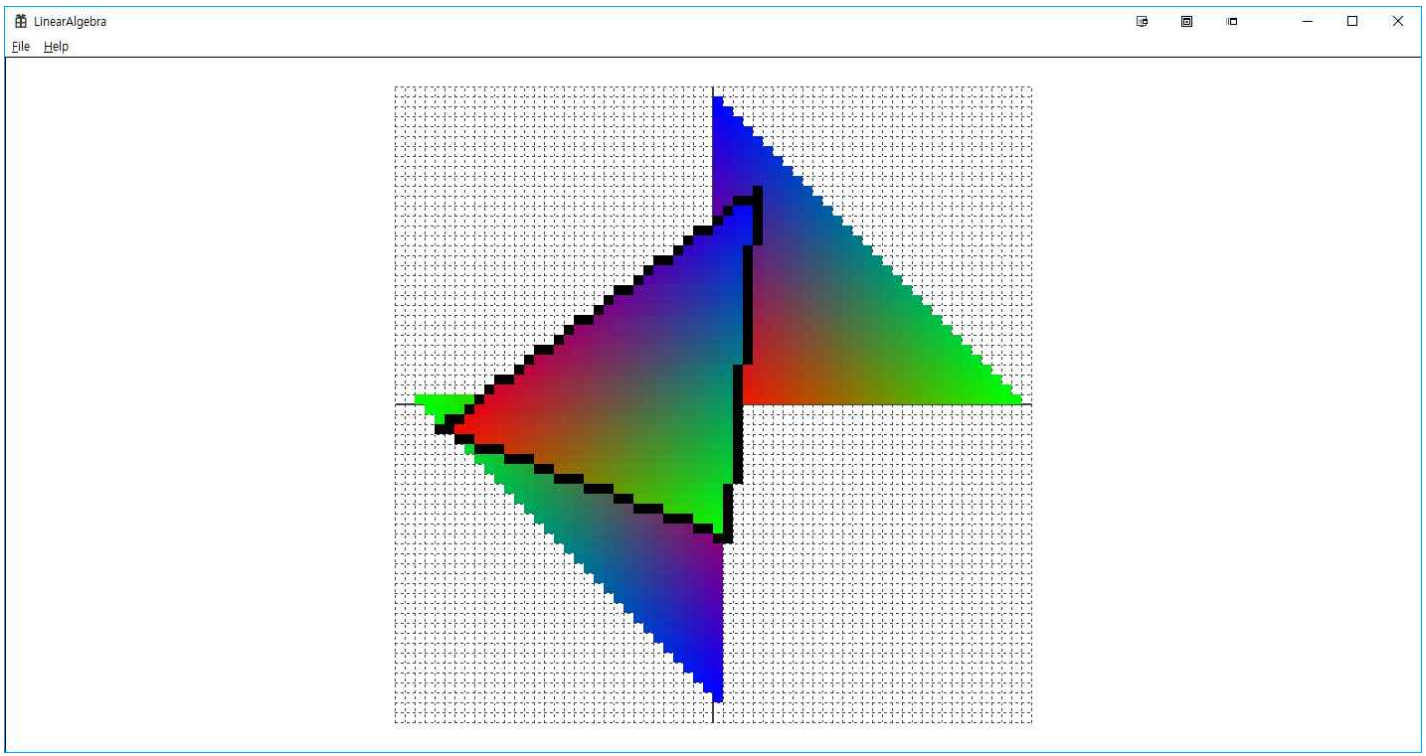
[Fig] Shows all tip pixels

[Fig] Final result of filled polygon

```
void KVectorUtil::FillTriangle(HDC hdc, int x1, int y1, KRgb const col1
    , int x2, int y2, KRgb const col2
    , int x3, int y3, KRgb const col3)

{
    std::set<ScannedResult> scanned_pnts; // sorted in y val, and in x val if y val the same.
    ScanLineSegment(hdc, x1, y1, col1, x2, y2, col2, &scanned_pnts);
    ScanLineSegment(hdc, x2, y2, col2, x3, y3, col3, &scanned_pnts);
    ScanLineSegment(hdc, x3, y3, col3, x1, y1, col1, &scanned_pnts);

    int cur_yval = INT_MIN;// vReso / 2; // initialize to an invalid value.
    std::set<ScannedResult> same_yval; // of the scanned result.
    int dbgCnt = 0;

    for (std::set<ScannedResult>::iterator it = scanned_pnts.begin(); it != scanned_pnts.end(); ++it)
    {
        int y = it->y;
        if (y != cur_yval)
        {
            if (same_yval.size())
            {
                std::set<ScannedResult>::iterator it1 = same_yval.begin(), it2 = --same_yval.end();
                ScanLineSegment(hdc, it1->x, cur_yval, it1->col,
                    it2->x, cur_yval, it2->col, nullptr);
```

```
#ifdef _DEBUG
                if (dbgCnt == g_idebug)
                    break;
                dbgCnt += 1;
#endif

                same_yval.clear();
            }
            cur_yval = y;
        }
        same_yval.insert(*it);
    }
    if (same_yval.size())
    {
        std::set<ScannedResult>::iterator it1 = same_yval.begin(), it2 = --same_yval.end();
        ScanLineSegment(hdc, it1->x, cur_yval, it1->col
            , it2->x, cur_yval, it2->col, nullptr);
    }
}
```

Result

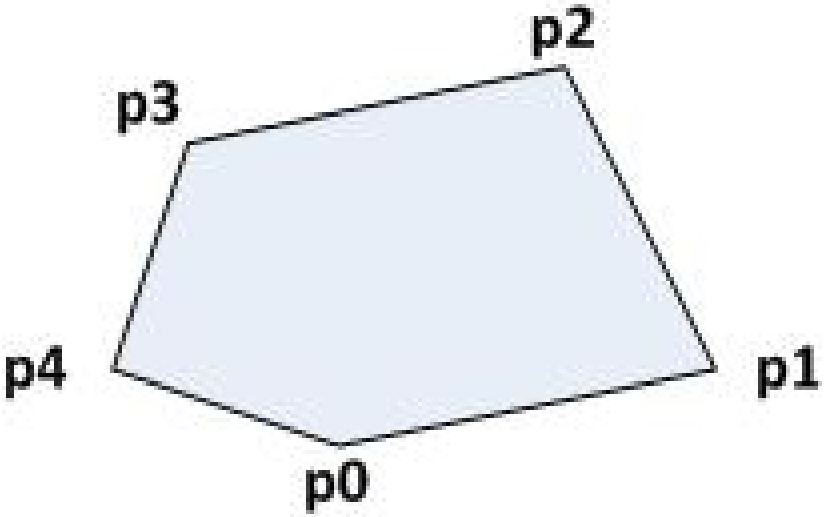


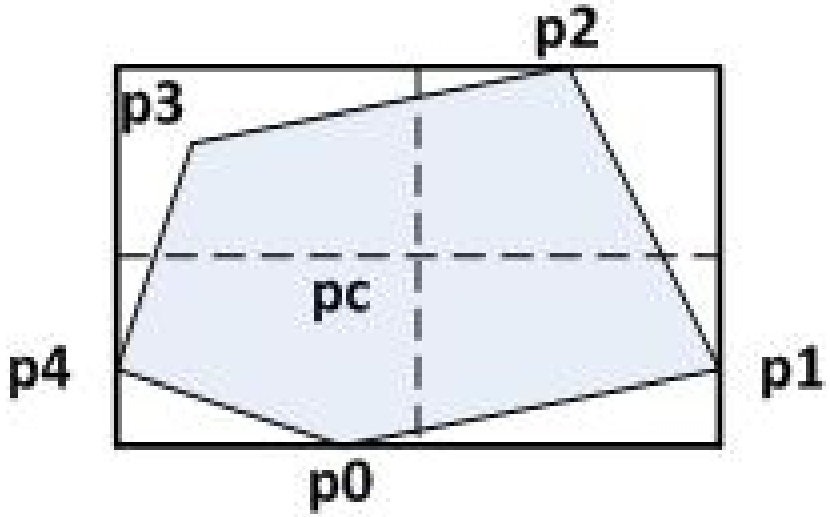
[Fig] DrawFilledTriangle

Practice: Implement DrawFilled2DPolygon()

Define class KPolygon2D

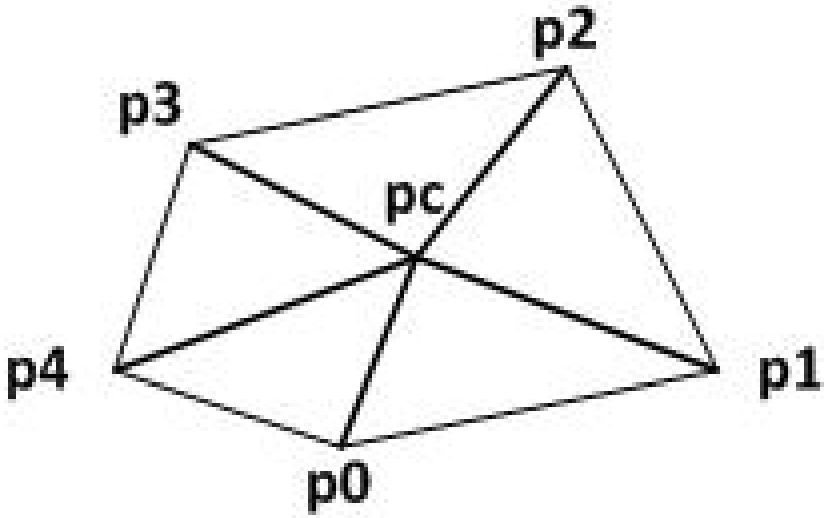
std::vector<KVector2> points;





Find the center of the bounding box

Triangulation, then apply FillTriangle()



[문서의 끝]