```
///////////////////////////////////////////////////////////////////////////////
// Demo: CPP-02.09D - Effective C++  (Maximum Call Version)                 //
///////////////////////////////////////////////////////////////////////////////

#include <iostream>
using namespace std;



class CPoint
///////////////////////////////////////////////////////////////////////////////
{
private:
   static int iCtorCount;
   static int iDtorCount;
   static int iCopyCount;
   static int iAssignCount;

public:
   int iX;
   int iY;

public:
   CPoint() : iX(0), iY(0) {iCtorCount ++;}
   CPoint(int x, int y) : iX(x), iY(y) {iCtorCount++;}
   CPoint(const CPoint& Source);
   CPoint& operator=(const CPoint& Source);
   ~CPoint() {iDtorCount++;}

   static void listCount(void)
   {
       cout << "--> Call Count: "
       << iCtorCount + iDtorCount + iCopyCount + iAssignCount
       << " (Ctor: " << iCtorCount
       << ", Dtor: " << iDtorCount
       << ", Copy: " << iCopyCount
       << ", Assign: " << iAssignCount << ")" << endl;
   }
};
// class: CPoint /////////////////////////////////////////////////////////////




// define and init static data members ///////////////////////////////////////
int CPoint::iCtorCount = 0;
int CPoint::iCopyCount = 0;
int CPoint::iDtorCount = 0;
int CPoint::iAssignCount = 0;



// point copy constructor ////////////////////////////////////////////////////
CPoint::CPoint(const CPoint& Source)
{
    *this = Source;
    iCopyCount++;
}
///////////////////////////////////////////////////////////////////////////////



// point assignment operator /////////////////////////////////////////////////
CPoint& CPoint::operator=(const CPoint& Source)
{
    iX = Source.iX;
    iY = Source.iY;
    iAssignCount++;
    return *this;
}
///////////////////////////////////////////////////////////////////////////////

```

```cpp


class CLine
///////////////////////////////////////////////////////////////////////////////
{
private:
   CPoint P1;
   CPoint P2;

public:
   CLine(const CPoint p1, const CPoint p2);
   void List(void);
};
// class: CLine ///////////////////////////////////////////////////////////////



CLine::CLine(const CPoint p1, const CPoint p2)
///////////////////////////////////////////////////////////////////////////////
{
   P1 = p1;
   P2 = p2;
}
///////////////////////////////////////////////////////////////////////////////



void CLine::List(void)
///////////////////////////////////////////////////////////////////////////////
{
   cout << "Line Object:" << endl;
   cout << " P1: iX = " << P1.iX << "  iY = " << P1.iY << endl;
   cout << " P2: iX = " << P2.iX << "  iY = " << P2.iY << endl << endl;
}
///////////////////////////////////////////////////////////////////////////////



int main(void)
///////////////////////////////////////////////////////////////////////////////
{
    // show initial call count
    CPoint::listCount();
    cout << endl;

    // definitions
    CLine* pL1;
    CPoint P1;
    CPoint P2(30, 30);

    // dynamic instantiation of a line object
    pL1 = new CLine(P1, P2);

    // list values of line
    pL1->List();

    // show final call count
    CPoint::listCount();
    cout << endl;

    // delete dynamic line object
    delete pL1;

    return 0;
}

```