EXHIBIT F

Source Code Exemplar

```
// Hello1.cpp

#include "stdafx.h"
#define NUM 5
static int notLong;

static int imaglobal;
int meToo = 2;
int meMid = 4;
int meThree = 3;


enum OP { ORIGIN, MAKERELOCS, DEFBYTE, REL32, ABS32, LAST_OP };

OP  opArray[] = {ORIGIN, MAKERELOCS, DEFBYTE, REL32, ABS32, LAST_OP};

void cleanup () {
        for ( int i = notLong; i > 0; i-- ) {
                printf("The End\n");
        }
        return;
}

OP getOP (int start ) {
        int remain = start % notLong;
        return opArray[remain];
}

int _tmain(int argc, _TCHAR* argv[])
{
        int notMe = NUM;
        bool notDone = true;
        imaglobal = 6;
        notLong = 6;

        while ( notDone ) {
                OP result = getOP(imaglobal);
                switch (result) {

                        case ORIGIN:
                                printf("ORIGIN\n");
                                imaglobal += 2;
                                break;
                        case MAKERELOCS:
                                printf("RELOC\n");

                                if ( meToo == 2 ) {
                                        meToo = 0;

                                        notDone = false;
                                }
                                imaglobal += meToo;
                                break;
                        case DEFBYTE:
                                imaglobal += notMe;
                                while ( notMe > 0 ) {
                                        printf("DEFBYTE\n");
                                        notMe--;

                                }
                                break;
                        case REL32:
                                printf("REL32\n");
                                imaglobal = meMid;
                                break;
                        case ABS32:
                                printf("ABS32\n");
                                imaglobal++;
                                break;
```

```
| // Hello2.cpp

  #include "stdafx.h"
  #define NUM 5
| static int tooShort = 4;
> static int notLong = 6;
  static int imaglobal;
  int meToo = 2;
<
  int meThree = 3;
> int meMid = 4;

  enum OP { ORIGIN, MAKERELOCS, DEFBYTE, REL32, ABS32, LAST_OP };
<
  OP  opArray[] = {ORIGIN, MAKERELOCS, DEFBYTE, REL32, ABS32, LAST_OP};

  void cleanup () {
          for ( int i = notLong; i > 0; i-- ) {
|                 printf("The End %d\n",i);
          }
          return;
  }
>
  OP getOP (int start ) {
          int remain = start % notLong;
          return opArray[remain];
  }

  int _tmain(int argc, _TCHAR* argv[])
  {
<
<
          imaglobal = 6;
|         bool notDone = true;
|         int origVal = imaglobal;
          while ( notDone ) {
|                 OP op = getOP(imaglobal);
|                 int norMe = NUM;
>                 switch (op) {
                          case ORIGIN:
                                  printf("ORIGIN\n");
                                  imaglobal += 2;
                                  break;
                          case MAKERELOCS:
                                  printf("RELOC\n");
                                  imaglobal += meToo;
>                                 if ( meToo == 2 ) {
                                          meToo = 0;
>                                 } else {
                                          notDone = false;
                                  }
<
                                  break;
                          case DEFBYTE:
|                                 while ( norMe > 0 ) {
<
                                          printf("DEFBYTE\n");
|                                         imaglobal++;
>                                         norMe--;
                                  }
                                  break;
                          case REL32:
                                  printf("REL32\n");
                                  imaglobal = meMid;
                                  break;
                          case ABS32:
                                  printf("ABS32\n");
                                  imaglobal++;
                                  break;
```

```
                        case LAST_OP:
                                printf("LAST\n");
                                imaglobal++;
                                if ( imaglobal < 20 ) continue;

                                notDone = false;
                                break;
                }
        }
    cleanup();
        return 0;
}
```

```
>
                case LAST_OP:
                        printf("LAST\n");
                        imaglobal++;
                        if ( imaglobal < 20 ) continue;
                        imaglobal = origVal;
                        notDone = false;
                        break;
                }
        }
    cleanup();
        return 0;
}
```