附録

- VisualC++ 2005以前で動作させる場合
- Quick Cライブラリ
- Turbo Cライブラリ
- PC-98用ライブラリ (mglib.h)

VisualC++2005以前で動作させる場合

本書プログラムをVisual C++2005以前にて動作させる場合，以下の点に注意すれば，そのまま動作させることができます．

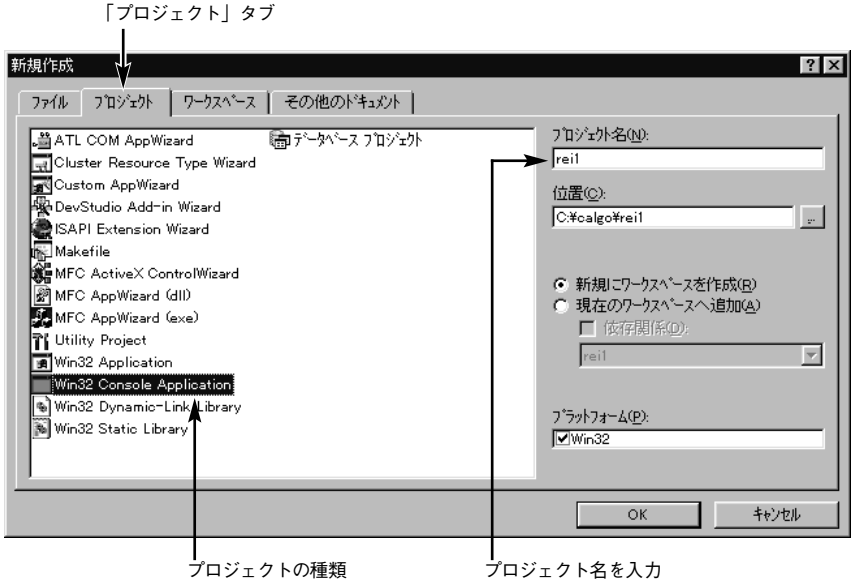
この附録の内容は第2版改訂時の附録の内容を一部変更したものです．

1．グラフィックスを扱わないプログラム

Visual C++のプロジェクトととして「Win32 Console Application」を選択すると，MS-DOSプロンプト（コンソール）上で動作するC言語アプリケーションを作ることができます。

本書の**例題1**をVisual C++の「Win32 Console Application」として作成する手順を以下に示します．

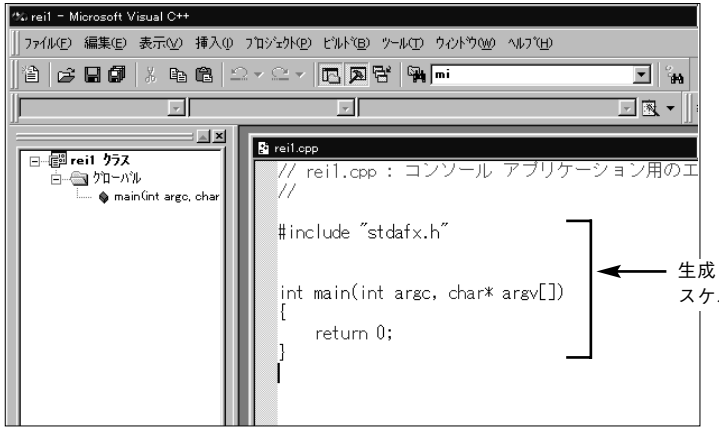
①［ファイル(F)］－［新規作成(N)］メニューで開くダイアログの［プロジェクト］タブを選択し，「Win32 Console Application」を選択し，プロジェクト名に「rei1」と入力する．



②アプリケーションの種類を［単純アプリケーション(S)］にする．

③スケルトンの生成

①，②の設定で以下のようなスケルトン（骨格）が生成されます．



【注】UNIX，Windowsなどのマルチタスクでは，プログラムの終了でOSにエラー情報などの戻り値を返すようにするため，main関数の型を「`int`」とし，「`return 0;`」でOSに戻り値を返します．通常，戻り値の意味は0なら正常終了，非0ならエラーを表します．

④プログラムの記述

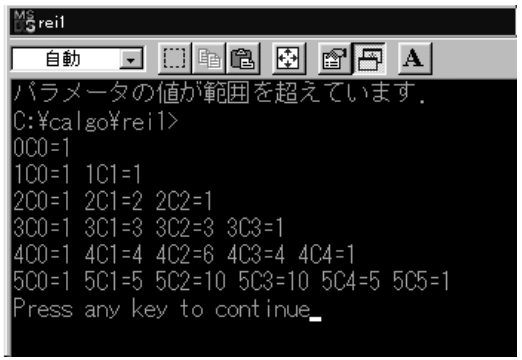
③で作成されたスケルトンにプログラム本体を記述します．

プログラム

```
// rei1.cpp : コンソール アプリケーション用のエントリ ポイントの定義
//

#include "stdafx.h"

#include <stdio.h>

long combi(int,int);

int main(int argc, char* argv[])
{
    int n,r;
    for (n=0;n<=5;n++){
        for (r=0;r<=n;r++)
            printf("%dC%d=%ld ",n,r,combi(n,r));
        printf("¥n");
    }
    return 0;
}
long combi(int n,int r)
{
    int i;
    long p=1;

    for (i=1;i<=r;i++)
        p=p*(n-i+1)/i;
    return p;
}
```

ユーザが記述したプログラム

⑤［ビルド(B)］－［実行］メニューで実行



【注意点】

1部バージョンで次のような不具合がある場合があります．

◎scanfに^Zを入力するプログラム

データ終わりの通知

scanf("%s",…)の%s書式でデータの入力を行っている場合のデータ終了マークは^Zを2回続けます．getsを用いれば^Zは1回で良いです．

改行処理

scanfに^Zを入力し，ループから抜け，その直後にコンソール出力を行う形式のプログラムは，コンソール出力の前に，printf("¥n");を行い改行して下さい．改行を行っておかないと，最初の行が表示されません．
たとえば，例題25のプログラムは次のようになります．

```
int main(int argc, char* argv[])
{
    int n;
    char a[12],b[12];
    printf("名前　電話番号¥n");
    while (scanf("%s %s",a,b)!=EOF){ ←― ^Z^Zで終了
        n=hash(a);
        strcpy(dat[n].name,a);
        strcpy(dat[n].telnum,b);
    }
    rewind(stdin);
    printf("¥n"); ←――― ここに改行コードを記述
    printf("検索するデータを入力してください¥n");
    while (scanf("%s",a)!=EOF){
        n=hash(a);
        printf("%15s%15s¥n",dat[n].name,dat[n].telnum);
    }
    return 0;
}
```

該当プログラム

Rei25，Dr25，Rei34，Dr34_1，Dr34_2，Rei35，Dr35_1，Dr35_2，Rei36，Dr36，Rei37，Dr37，Rei40，Dr40，Rei41，Dr41，Dr43，Rei44，Rei45，Rei46，Dr46，Rei47，Dr47，Rei48，Dr48_1，Dr48_2

◎日本語入力

Visual C++の「Win32 Console Application」において，windows95などの一部環境でコンソールから日本語入力できない場合があります．

2．グラフィックスを扱うプログラム

Windowsでグラフィックスを行うにはMFC（Microsoft Foundataion Class）を利用するのが最も簡単です．

しかし，MFCを使う場合はクラスの知識が必要になります．そこで，本書のプログラムをクラスを意識せずにそのままVisual C++のMFプロジェクト上で動作させるためのmfcglib.hを作成しました．

このmfcglib.hをVisual C++のIncludeフォルダに置いて下さい．

●mfcglib.h（VisualC++　MFCアプリケーション版）

```
/*
 * ----------------------------------------
 *      基本グラフィックスライブラリ          *
 *     VisualC++, Visual C++ 2005/2008     *
 *       MFC MFCアプリケーション版          *
 * ----------------------------------------
 */
#include <math.h>

CDC* gpDC;              /* デバイスコンテキスト */
CPen mypen(PS_SOLID,1,RGB(0,0,255)); /* 描画ペン */

double WX1,WY1,WX2,WY2,      /* ワールド座標 */
       VX1,VY1,VX2,VY2       /* ビュー座標   */
       FACTX,FACTY,          /* スケール     */
       ANGLE,                /* 現在角       */
       LPX,LPY;              /* 現在位置     */

void window(double x1,double y1,double x2,double y2)
{
    WX1=x1; WY1=y1; WX2=x2; WY2=y2;
    FACTX=(VX2-VX1)/(WX2-WX1);
    FACTY=(VY2-VY1)/(WY2-WY1);
}
void view(double x1,double y1,double x2,double y2)
{
    CRgn clip;  /* クリップ領域の指定 */
    clip.CreateRectRgn((int)x1,(int)y1,(int)x2+1,(int)y2+1);
    gpDC->SelectObject(clip);
    VX1=x1; VY1=y1; VX2=x2; VY2=y2;
    FACTX=(VX2-VX1)/(WX2-WX1);
    FACTY=(VY2-VY1)/(WY2-WY1);
}
void Ginit(CDC* p)
{
    gpDC=p;
    gpDC->SetMapMode(MM_ANISOTROPIC);
    pDC->SelectObject(&mypen);
        // 後処理も行うなら各描画処理のところで以下のようにする
        // CPen* oldp=gpDC->SelectObject(&mypen);
        // 描画処理
        // gpDC->SelectObject(oldp);
    window(0,0,639,399);
    view(0,0,639,399);
}
void cls(void)
{
    CRect r;
    gpDC->GetClipBox(&r);
    gpDC->FillSolidRect(r,gpDC->GetBkColor());
}
void line(double x1,double y1,double x2,double y2)
{
    int px1,py1,px2,py2;
    px1=(int)((x1-WX1)*FACTX+VX1);
    py1=(int)((WY2-y1)*FACTY+VY1);
    px2=(int)((x2-WX1)*FACTX+VX1);
    py2=(int)((WY2-y2)*FACTY+VY1);
    gpDC->MoveTo(px1,py1);
    gpDC->LineTo(px2,py2);
    LPX=x2;LPY=y2;
}
void pset(double x,double y)
{
    int px,py;
    px=(int)((x-WX1)*FACTX);
    py=(int)((WY2-y)*FACTY);
    gpDC->SetPixel(px,py,RGB(0,0,255));
    LPX=x;LPY=y;
}
void move(double l)
{
    double x,y,rd=3.1415927/180;
    x=l*cos(rd*ANGLE);y=l*sin(rd*ANGLE);
    line(LPX,LPY,LPX+x,LPY+y);
}
void moveto(double x,double y)
{
    line(LPX,LPY,x,y);
}
void setpoint(double x,double y)
{
    LPX=x;LPY=y;
```

```
}

#define setangle(a) ANGLE=(double)(a)
#define turn(a) ANGLE=fmod(ANGLE+(a),360.0)

#define ginit() Ginit(pDC)
```

【注意点】

◎ginitの位置

GinitはOnDrawイベントハンドラのpDC引数をマクロ展開しているので必ずOnDrawイベントハンドラの中に置かなければならない.

◎描画色

青に設定してあるが，変更したい場合は以下のRGB(0,0,255)の値を変える.

```
CPen mypen(PS_SOLID,1,RGB(0,0,255));
```

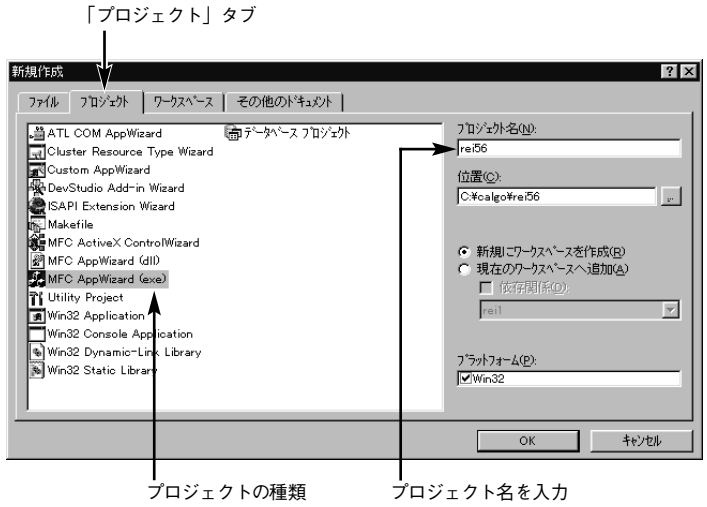
◎windowマクロ

引数の型はdouble型になっているが、整数の実引数を与えた場合に警告がでる場合は、window関数を例にすると以下のようなマクロwindowと実体関数Windowを作るとよい。

```
#define window(x1,y1,x2,y2) ¥
   Window((double)(x1),(double)(y1),(double)(x2),(double)(y2))
void Window(double x1,double y1,double x2,double y2)
{

}
```

本書の例題56をVisual C++の「MFC Application」として作成する手順を以下に示します.

①[ファイル(F)]－[新規作成(N)]メニューで開くダイアログの[プロジェクト]タブを選択し，「MFC AppWizard（exe)」を選択し，プロジェクト名に「rei56」と入力する.



②アプリケーションの種類を[SDI(S)]にする.

SDIはSingle Document Interfaceの略で，単一のドキュメントしか扱えないプログラム（たとえばメモ帳のような）です.

③スケルトンの生成

①，②の設定で以下のようなスケルトン（骨格）プログラムが生成されます．
プログラムは「CRei56View」の「OnDraw」関数内に記述します．

④プログラムの記述

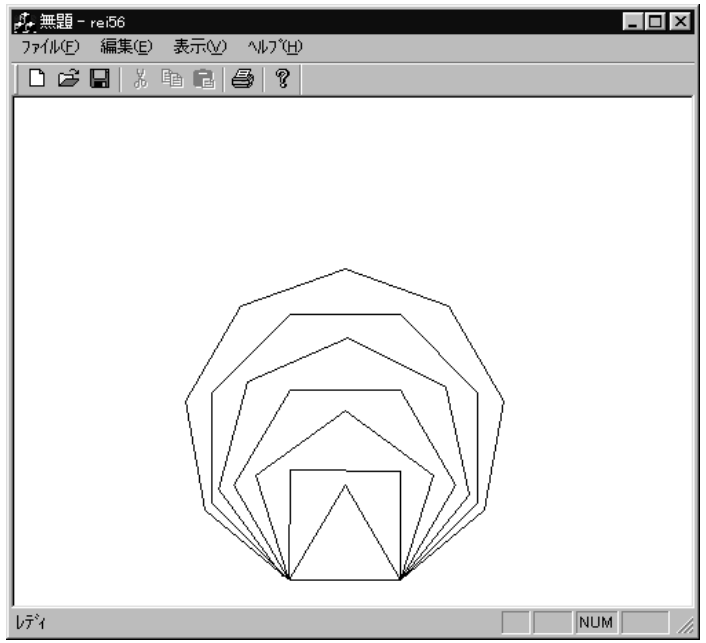
③で作成されたスケルトンプログラムの「CRei56View」クラスの「OnDraw」関数の中にプログラム本体を記述します．

プログラム

```
#include "mfcglib.h"          ← グラフィックスライブラリのインクルード
                              ← ユーザ関数のある場合はここに記述
void CRei56View::OnDraw(CDC* pDC)
{
    CRei56Doc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    // TODO: この場所にネイティブ データ用の描画コードを追加します。
    int j,n;

    ginit();cls();

    for (n=3;n<=9;n++){
        setpoint(200,50);
        setangle(0);
        for (j=0;j<n;j++){
            move(80);
            turn(360/n);
        }
    }
}
```

main 関数内のコードを記述

⑤［ビルド(B)］－［実行］メニューで実行



Quick Cライブラリ

Quick Cのグラフィックス・ライブラリはviewとwindowをsetviewportとsetwindowという関数でサポートしている．また，setwindowのオプションによりy方向を逆転させることができる．ただし，この場合のウィンドウの指定は(x1,-y2,x2,-y1)のようにy1とy2の値を入れ換え，負号を逆にする必要がある．

プログラム

```
/*
 * ---------------------------------------
 *      基本グラフィックス・ライブラリ     *
 *          （Ｑｕｉｃｋ Ｃ版）            *
 * ---------------------------------------
 */

#include <graph.h>
#include <math.h>


#define cls() ¥
        _clearscreen(_GCLEARSCREEN)
#define window(x1,y1,x2,y2) ¥
        _setwindow(1,(double)(x1),-(double)(y2),(double)
        (x2),-(double)(y1))
#define view(x1,y1,x2,y2) ¥
        _setviewport((short)(x1),(short)(y1),(short)(x2),
        (short)(y2))
#define line(x1,y1,x2,y2) ¥
        Line((double)(x1),(double)(y1),(double)(x2),
        (double)(y2))
#define pset(x,y) ¥
        _setpixel_w((double)(x),(double)(y))
#define setpoint(x,y) ¥
        SetPoint((double)(x),(double)(y))
#define setangle(a) ¥
        ANGLE=(double)(a)
#define turn(a) ¥
        ANGLE=fmod(ANGLE+(a),360.0)
#define move(l) ¥
        Move((double)(l))
#define moveto(x,y) ¥
        MoveTo((double)(x),(double)(y))
#define moverel(x,y) ¥
        MoveTo((double)(LPX+(x)),(double)(LPY+(y)))

double ANGLE,           /* 現在角    */
       LPX,LPY;         /* 現在位置  */
```

```
void ginit(void)
{
    _setvideomode(_98RESSCOLOR);
    window(0,0,639,399);
    view(0,0,639,399);
}
void Line(double x1,double y1,double x2,double y2)
{
    _moveto_w(x1,y1); _lineto_w(x2,y2);
}
void Move(double l)
{
    double x,y,rd=3.14159/180;
    x=l*cos(rd*ANGLE); y=l*sin(rd*ANGLE);
    _lineto_w(LPX+x,LPY+y);
    LPX=LPX+x;LPY=LPY+y;
}
void MoveTo(double x,double y)
{
    _lineto_w(x,y);
    LPX=x; LPY=y;
}
void SetPoint(double x,double y)
{
    _moveto_w(x,y);
    LPX=x; LPY=y;
}
```

Turbo Cライブラリ

Turbo Cのグラフィックス・ライブラリはwindowをサポートせず，描画座標は整数型である．

view機能はsetviewport関数によりサポートしており，これによりビューポートの原点への平行移動が行われるため，②式（本紙P.365）は

$$x' = (x - WX1) * FACTX$$
$$y' = (WY2 - y) * FACTY$$

となる．

プログラム

```
/*
 * ------------------------------------------
 *         基本グラフィックス・ライブラリ          *
 *             （ＴｕｒｂｏＣ対応版）              *
 * ------------------------------------------
 */
#include <stdio.h>
#include <graphics.h>
#include <process.h>
#include <math.h>

#define cls() ¥
        clearviewport()
#define pset(x,y) ¥
        Line((x),(y),(x),(y))
#define window(x1,y1,x2,y2) ¥
        Window((double)(x1),(double)(y1),(double)(x2),(double)
        (y2))
#define view(x1,y1,x2,y2) ¥
        View((int)(x1),(int)(y1),(int)(x2),(int)(y2))
#define setpoint(x,y) ¥
        SetPoint((double)(x),(double)(y))
#define setangle(a) ¥
        ANGLE=(double)(a)
#define turn(a) ¥
        ANGLE=fmod(ANGLE+(a),360.0)
#define move(l) ¥
        Move((double)(l))
#define moveto(x1,y1)   ¥
        MoveTo((double)(x1),(double)(y1))
#define moverel(x,y) ¥
        MoveTo((double)(LPX+(x)),(double)(LPY+(y)))
```

```
double WX1,WY1,WX2,WY2,         /* ワールド座標 */
       VX1,VY1,VX2,VY2,         /* ビュー座標   */
       FACTX,FACTY,             /* スケール     */
       ANGLE,                   /* 現在角       */
       LPX,LPY,                 /* 現在位置     */
       rd=3.1415927/180;

void ginit(void);
void View(int,int,int,int);
void Window(double,double,double,double);
void Line(double,double,double,double);

void ginit(void)
{
    int gerr,gmode,gdriver=DETECT;
    initgraph(&gdriver,&gmode,"a:¥¥");
    if ((gerr=graphresult())!=grOk){
        printf("Error : %s¥n",grapherrormsg(gerr));
        exit(1);
    }
    window(0,0,639,399);
    view(0,0,639,399);
}
void View(int x1,int y1,int x2,int y2)
{
    setviewport(x1,y1,x2,y2,1);
    VX1=(double)x1; VY1=(double)y1; VX2=(double)x2; VY2=
    (double)y2;
    FACTX=(VX2-VX1)/(WX2-WX1);
    FACTY=(VY2-VY1)/(WY2-WY1);
}
void Window(double x1,double y1,double x2,double y2)
{
    WX1=x1; WY1=y1; WX2=x2; WY2=y2;
    FACTX=(VX2-VX1)/(WX2-WX1);
    FACTY=(VY2-VY1)/(WY2-WY1);
}
void Line(double x1,double y1,double x2,double y2)
{
    int px1,py1,px2,py2;
    px1=(int)((x1-WX1)*FACTX);
    py1=(int)((WY2-y1)*FACTY);
    px2=(int)((x2-WX1)*FACTX);
    py2=(int)((WY2-y2)*FACTY);
    line(px1,py1,px2,py2);
    LPX=x2;  LPY=y2;
}
void SetPoint(double lpx,double lpy)
{
    LPX=lpx; LPY=lpy;
}
void Move(double l)
{
```

```
    double x,y;
    x=l*cos(rd*ANGLE);  y=l*sin(rd*ANGLE);
    Line(LPX,LPY,LPX+x,LPY+y);
}
void MoveTo(double x,double y)
{
    Line(LPX,LPY,x,y);
}

#define line(x1,y1,x2,y2)¥
        Line((double)x1,(double)y1,(double)x2,(double)y2)
```


PC-98用ライブラリ(mglib.h)

●mglib.h

```
/*
 ----------------------------------------
 *    基本グラフィック・ライブラリ          *
 *      （ＰＣ－９８グラフＬＩＯを利用）    *
 ----------------------------------------
 */

#include <stdio.h>
#include <process.h>
#include <math.h>
#include <dos.h>
#include <malloc.h>

#define window(x1,y1,x2,y2) ¥
        Window((double)(x1),(double)(y1),(double)(x2),
        (double)(y2))
#define view(x1,y1,x2,y2) ¥
        View((int)(x1),(int)(y1),(int)(x2),(int)(y2))
#define line(x1,y1,x2,y2) ¥
        Line((double)(x1),(double)(y1),(double)(x2),
        (double)(y2))
#define pset(x,y) ¥
        Pset((double)(x),(double)(y))
#define setpoint(x,y) ¥
        SetPoint((double)(x),(double)(y))
#define setangle(a) ¥
        ANGLE=(double)(a)
#define turn(a) ¥
        ANGLE=fmod(ANGLE+(a),360.0)
#define move(l) ¥
        Move((double)(l))
#define moveto(x,y) ¥
        MoveTo((double)(x),(double)(y))
#define moverel(x,y) ¥
        MoveTo((double)(LPX+(x)),(double)(LPY+(y))

void setb(int,int);         /* 関数プロトタイプ宣言 */
void setw(int,int);
void lio(int);
void ginit(void);
void View(int,int,int,int);
void Window(double,double,double,double);
void screen(int,int,int,int);
void cls(void);
void Pset(double,double);
void Line(double,double,double,double);
void Move(double);
void SetPoint(double,double);
```

```
void MoveTo(double,double);

union REGS inregs,outregs;    /* レジスタ共用体 */
struct SREGS segregs;
unsigned DSEG,GSEG;

double WX1,WY1,WX2,WY2,       /* ワールド座標 */
       VX1,VY1,VX2,VY2,       /* ビュー座標   */
       FACTX,FACTY,           /* スケール     */
       ANGLE,                 /* 現在角       */
       LPX,LPY,               /* 現在位置     */
       rd=3.1415927/180;

unsigned IRET=0xcfcf;         /* ｉｒｅｔコード */

void setb(int adr,int dat)    /* バイトデータの設定 */
{
    movedata(DSEG,(int)&dat,GSEG,adr,1);
}
void setw(int adr,int dat)    /* ワードデータの設定 */
{
    movedata(DSEG,(int)&dat,GSEG,adr,2);
}
void lio(int intno)           /* ｇｌｉｏ割り込み */
{
    inregs.x.bx=0;
    segregs.ds=GSEG;
    int86x(intno,&inregs,&outregs,&segregs);
}
void ginit(void)              /* グラフィック画面の初期化 */
{
    int i;
    unsigned WORK;
    char *gwork;
    if ((gwork=malloc(0x1400))==0){    /* ｇｌｉｏ作業域 */
        printf("メモリ不足¥n");
        exit(1);
    }
    segread(&segregs); DSEG=segregs.ds;
    GSEG=DSEG+(int)gwork/16;        /* 作業域のセグメント値 */

    WORK=0xf990;
    for (i=0;i<16;i++){             /* ｇｌｉｏベクタの設定 */
        movedata(0xf990,6+i*4,0x0000,0xa0*4+i*4,2);
        movedata(DSEG,(int)&WORK,0x0000,0xa0*4+i*4+2,2);
    }
    movedata(0xf990,6+16*4,0x0000,0xce*4,2);
    WORK=(unsigned)&IRET;
    movedata(DSEG,(int)&WORK,0x0000,0xc5*4,2);
    movedata(DSEG,(int)&DSEG,0x0000,0xc5*4+2,2);
    lio(0xa0);
    screen(3,0,0,1);
}
```

```
void View(int x1,int y1,int x2,int y2)
{
    setw(0,x1);         /* 左上ｘ座標 */
    setw(2,399-y2);     /* 左上ｙ座標 */
    setw(4,x2);         /* 右下ｘ座標 */
    setw(6,399-y1);     /* 右下ｙ座標 */
    setb(8,0xff);       /* 領域色 */
    setb(9,0xff);       /* 境界色 */
    lio(0xa2);
    VX1=x1; VY1=399-y2; VX2=x2; VY2=399-y1;
    FACTX=(VX2-VX1)/(WX2-WX1);
    FACTY=(VY2-VY1)/(WY2-WY1);
}
void Window(double x1,double y1,double x2,double y2)
{
    WX1=x1; WY1=y1; WX2=x2; WY2=y2;
    FACTX=(VX2-VX1)/(WX2-WX1);    /* ｘ方向の倍率 */
    FACTY=(VY2-VY1)/(WY2-WY1);    /* ｙ方向の倍率 */
}
void screen(int a,int b,int c,int d)
{
    setb(0,a);          /* 画面モード   */
    setb(1,b);          /* 画面スイッチ */
    setb(2,c);          /* アクテイブ   */
    setb(3,d);          /* デイスプレイ */
    lio(0xa1);
    WX1=WY1=VX1=VY1=0.0;
    WX2=VX2=639.0;
    if (a==0)
        WY2=VY2=199.0;
    else
        WY2=VY2=399.0;
    window(WX1,WY1,WX2,WY2);
    view(VX1,VY1,VX2,VY2);
}
void cls(void)
{
    printf("%c[2J",0x1b);    /* テキスト画面 */
    lio(0xa5);               /* グラフ画面   */
}
void Pset(double x,double y)
{
    setw(0,(int)((x-WX1)*FACTX+VX1));   /* ｘ座標 */
    setw(2,(int)((WY2-y)*FACTY+VY1));   /* ｙ座標 */
    setb(4,7);                          /* パレット番号 */
    lio(0xa6);
    LPX=x; LPY=y;
}
void Line(double x0,double y0,double x1,double y1)
{
    setw(0,(int)((x0-WX1)*FACTX+VX1));   /* 始点 */
    setw(2,(int)((WY2-y0)*FACTY+VY1));
    setw(4,(int)((x1-WX1)*FACTX+VX1));   /* 終点 */
```

```
    setw(6,(int)((WY2-y1)*FACTY+VY1));
    setb(8,7);                /*   パレット番号 */
    setw(9,0);                /*     描画コード */
    lio(0xa7);
    LPX=x1; LPY=y1;
}

void Move(double l)         /* 指定長の直線 */
{
    double x,y;
    x=l*cos(rd*ANGLE); y=l*sin(rd*ANGLE);
    line(LPX,LPY,LPX+x,LPY+y);
}
void SetPoint(double lpx,double lpy)  /* ＬＰ位置の設定 */
{
    LPX=lpx; LPY=lpy;
}
void MoveTo(double x,double y)    /* ＬＰから指定点への直線 */
{
    line(LPX,LPY,x,y);
}
```