Базаров С.Б., Баяковский Ю.М., Сейдалиева Ф.Ф., Скачков А.Ю.

Адаптация комплекса графических программ ГРАФОР в операционных системах WINDOWS и LINUX

Аннотация

Излагаются особенности использования пакета ГРАФОР – набора графических подпрограмм и функций, в ОС Windows 95/98/ME/NT/2000 (при использовании транслятора Compaq Visual Fortran 6.X) и Linux.

S.B.Bazarov, Yu.M.Bayakovsky, F.F.Seidalieva, A.Yu.Skachkov

Adaptation of software library GRAFOR for Windows and Linux.

Abstract

The application of GRAFOR (GRAphical FORtran – the set of routines for graphics) in the Microsoft® Visual Studio and in OS Linux is considered.

Работа выполнена при поддержке РФТР (проект 85/2001).

Содержание.

Введение

В данной работе описаны изменения и усовершенствования, которые были внесены в пакет ГРАФОР с момента выхода работы [♠], а также реализация пакета в набирающей популярность операционной системе Linux. Приведены описания вызовов подпрограмм пакета из C/C++. Описан формат файлов на языке HP-GL и программа для их просмотра.

Формат файлов HP-GL, используемый в пакете ГРАФОР

Для повышения мобильности пакета ГРАФОР было сделано следующее. Поскольку средства непосредственного вывода изображения в различных операционных системах и/или средах различны, вывод изображения на экран осуществляется так: сначала программа, использующая пакет, записывает файл в подмножество языка HP-GL и файл закрывается, а затем вызывается подпрограмма, читающая этой файл, интерпретирующая команды HP-GL и выводящая соответствующую графику на экран. Заметим, что при этом вся “вычислительная часть” пакета (т.е. получение изображения, а ещё точнее – координат отрезков) остается неизменной, а подпрограмма,

[♠] Базаров С.Б., Баяковский Ю.М. Комплекс графических программ ГРАФОР в среде WINDOWS. Препринт ИПМ № 30. 2000г.

реализующая непосредственный вывод на дисплей, может использовать различные средства для этого вывода, а также может быть написана и на другом языке программирования.

Остановимся более подробно на языке описания данных HP-GL, созданным более четверти века назад (1976 г.) и поддерживаемом (наряду с более поздним стандартом HP-GL/2) многими производителями (как оборудования, в основном плоттеров и принтеров, так и программных средств, в первую очередь CAD-систем).

Язык использует только коды ASCII. Оператор языка обязательно включает мнемонический код операции (mnemonics), который состоит из двух прописных либо строчных букв, и несколько параметров, разделенных запятыми и/или пробелами. В качестве разделителя между мнемоническим кодом и первым параметром используется пробел, который может отсутствовать. Параметры могут быть необязательными. Если опущен какой либо из них, то должны отсутствовать и все параметры, следующие за ним.

Для разделения операторов используется точка с запятой, в случае отсутствия разделителя концом оператора считается начало следующего. Управляющие символы CR и LF при интерпретации игнорируются.

Задаваемые к качестве параметров в операторе числовые значения записываются в виде цепочек ASCII-символов. Допускается до пяти знаков дробной части. Если дробная часть отсутствует, десятичная точка может опускаться. В качестве параметров других типов допускаются также только цепочки символов. Целые значения типа clamped integer должны удовлетворять неравенству $(-32767) \leq \text{ClampedIntegerType} \leq 32767$, а вещественные $(-32767.9999) \leq \text{ClampedRealType} \leq 32767.9999$.

Таким образом, структура оператора (назовем его XX) выглядит следующим образом:

XX<параметр>…<разделитель><параметр;

Пример вариантов записи конкретного оператора:

“PDPU10,20” или “PD;PU10,20;” или “PD PU 10 20;”

Ядро языка HP-GL/2 содержит 55 операторов, которые должны поддерживаться всеми устройствами вывода изображения. Заметим, что даже при этом возможны случаи “разночтения” разными устройствами. Например, команда LT (LineType – тип

линии) в плоттере HP7475A воспринимает первый параметр, принимающий значения 0, 1, 2 , 3, 4, 5 и 6. А для плоттера HP7550A он может быть и отрицательным (0, ± 1, ± 2, ± 3, ± 4, ± 5, ± 6). К тому же в перьевых плоттерах карусели имеют различное количество гнезд ($\equiv$ цветов).

Таким образом, лучше выбрать и использовать только необходимый минимум команд, позволяющих реализовывать все графические операции. Приведем их краткое описание.

IN – Инициализация. Восстанавливает стандартные значения всех параметров устройства.

IP $ЛН_x$ $ЛН_y$ $ПВ_x$ $ПВ_y$ – определение значений масштабных точек (левого нижнего и правого верхнего углов чертежа).

SC X_{min}, X_{max}, Y_{min}, Y_{max} – задание системы координат пользователя.

SP N – установить цвет, определяемый номером.

PU X, Y – переместиться в указанную точку.

PD X, Y – провести линию в указанную точку.

Для наглядности приведем фрагмент файла, записываемого в результате работы пакета:

```
IN;
IP     0,  4019,  6028,     0;
SC     0,   300,     0,   200;
 SP
 1
 ;PU
 139
 139
 ;PD
 139
 20
 ;SP
 2
 ;PU
 139
 25
 ;PD
 278
 25
. . . . . .

;SP
 3
 ;PU
 265
 101
 ;PD
 267
 106
 ;SP0;
```

Программный модуль пакета ГРАФОР для записи файлов в формате HP-GL.

Создана подпрограмма пакета, осуществляющая запись результатов работы в файл, формат которого является подмножеством (используется минимум команд, достаточный для реализации всех графических элементов) языка HP-GL (Hewlett-Packard Graphic Language). Поскольку формат таких файлов чисто текстовый (plain ASCII), обеспечивается их передача на другие платформы и операционные системы.

Программный модуль отображения результатов работы ГРАФОРа в Windows.

Для повышения мобильности программ отображения (обычно эта часть программное обеспечения наиболее трудно переносится на другие платформы) их реализация в системах семейства Windows (95/98/ME/NT/2000) теперь осуществляется средствами OpenGL. В новой версии экранного драйвера она вызывается вместо подпрограммы, в которой отображение на экран осуществляется с помощью встроенных средств MS Fortran Power Station/Compaq Visual Fortran. При этом снимается ограничение на количество открываемых окон проекта.

Относительно старой версии также добавились возможности: более простой передачи изображения в буфер обмена, масштабирования и подстраивания под размер окна.

Взаимодействие подпрограмм, написанных на разных алгоритмических языках.

Также реализована отдельная динамическая библиотека HglView.dll, которая содержит функции, позволяющие просматривать HP-GL-файлы формата, описанного выше. Универсальность модуля позволяет вызывать его функции как из приложений, написанных на Фортране, так и на Си. HglView.dll создает собственное меню, через которое можно управлять окнами просмотра HP-GL файлов. Меню содержит следующие команды:

File	–>	Close	– Закрыть активное окно просмотра HP-GL файла
Edit	–>	Copy .hgl file to the Clipboard	– Скопировать содержимое активного окна в буфер обмена
View	–>	Size To Fit	– Изменить масштаб и размеры изображения в соответствии с реальными размерами HP-GL файла
	–>	Zoom In	– Увеличить размер изображения
	–>	Zoom Out	– Уменьшить размер изображения
Window	–>	Cascade	– Расположить окна с перекрытием
	–>	Tile	– Расположить окна без перекрытия
	–>	Close All	– Закрыть все окна просмотра HP-GL файлов

Список функций.

READHGLFILE	–	выбрать HPGL файл через стандартный OPEN диалог и показать его.
VIEWHGLFILE	–	показать заданный HPGL файл.
SIZETOFIT	–	изменить размеры окна HPGL изображения в соответствии с его реальным размером.
ZOOM	–	осуществить масштабирование HPGL изображения в сторону увеличения/уменьшения; масштабирование осуществляется по следующей шкале: ... 12,5% 25% 50% 100% 150% 200% 250% ...
ISHGLVIEW	–	определить, содержит ли заданное окно HPGL изображение.

Состав программного модуля.

1) HglView.dll - следует разместить либо в системной директории, либо в той, где будет находиться приложение, обращающееся к данному модулю.
2) HglView.lib - библиотека, необходимая для linker'a.
3) HglViewExt.h - файл, содержащий объявление функций; необходим для нормальной работы модулей на СИ.
4) Cview файлы - Пример работы с библиотекой на СИ.

Напомним, что подпрограммы пакета доступны при включении в проект файла **GRAFOR.OBJ,** который может находиться в произвольной директории на жёстком диске.

Вызов из Фортрана.

Для работы в ФОРТРАНЕ требуется Compaq Visual Fortran 6.X. Необходимо выполнить следующие шаги:

1) Создать проект QuickWin (multiple windows).
2) Добавить в созданный проект библиотеку HglView.lib .
3) Добавить в файл на ФОРТРАНЕ вызов нужной фукции.

Вызов из Си (Си++).

Для работы в СИ требуется наличие MS Visual C++ 6.0.

Необходимо выполнить следующие шаги:

1) Создать проект Win32 Application.
2) Добавить в созданный проект файл HglViewExt.h и библиотеку HglView.lib.
3) Создать обычное MDI приложение.

Пример Cview позволяет увидеть, что нужно добавить для работы с данной библиотекой.

Чтобы иметь возможность работать с меню, создаваемым для HPGL окна надо в процедуру обработки сообщения (message) WM_COMMAND

добавить следующие строки:

```
LONG    lResult = 0;
HWND    hChild;

hChild =(HWND)SendMessage(hwndMDIClient,WM_MDIGETACTIVE,0,(LPARAM)NULL);
if (ISHGLVIEW (hChild)) //  если handle принадлежит HPGL окну, то передать
{                       // команду меню на обработку процедуре HPGL окна
   lResult = SendMessage (hChild, WM_COMMAND, wParam, lParam);
   if (!lResult) return 0;
}
```

Описание функций.

__declspec (dllimport) extern BOOL __stdcall READHGLFILE (void);

С помощью этой функции пользователь может выбрать HPGL файл для просмотра через стандартное OPEN диалоговое окно

Аргументы: нет.
Возвращаемое значение: TRUE – в случае благополучного завершения,
FALSE – в случае неудачи.

__declspec (dllimport) extern BOOL __stdcall VIEWHGLFILE (LPINT *pnFile*);

Данная функция открывает для просмотра предварительно заданный HPGL файл

Аргументы: *pnFile* – указатель на номер HPGL файла. По умолчанию имя HPGL файла - HPxx.HGL, где xx ::= 1|2|...|99.
Возвращаемое значение: TRUE – в случае благополучного завершения,
FALSE – в случае неудачи.

__declspec (dllimport) extern void __stdcall SIZETOFIT (void);

Функция восстанавливает реальные размеры и масштаб HPGL изображения.

Аргументы: нет.
Возвращаемое значение: нет.

__declspec (dllimport) extern void __stdcall ZOOM (int *nDirection*);

Функция осуществляет масштабирование HPGL изображения; масштабирование осуществляется по следующей шкале: ... 12,5% 25% 50% 100% 150% 200% 250% ...

Аргументы:	*nDirection*	– направление "зуммирования":
		1 – в сторону увеличения
		– 1 – в сторону уменьшения
Возвращаемое значение:	нет.	

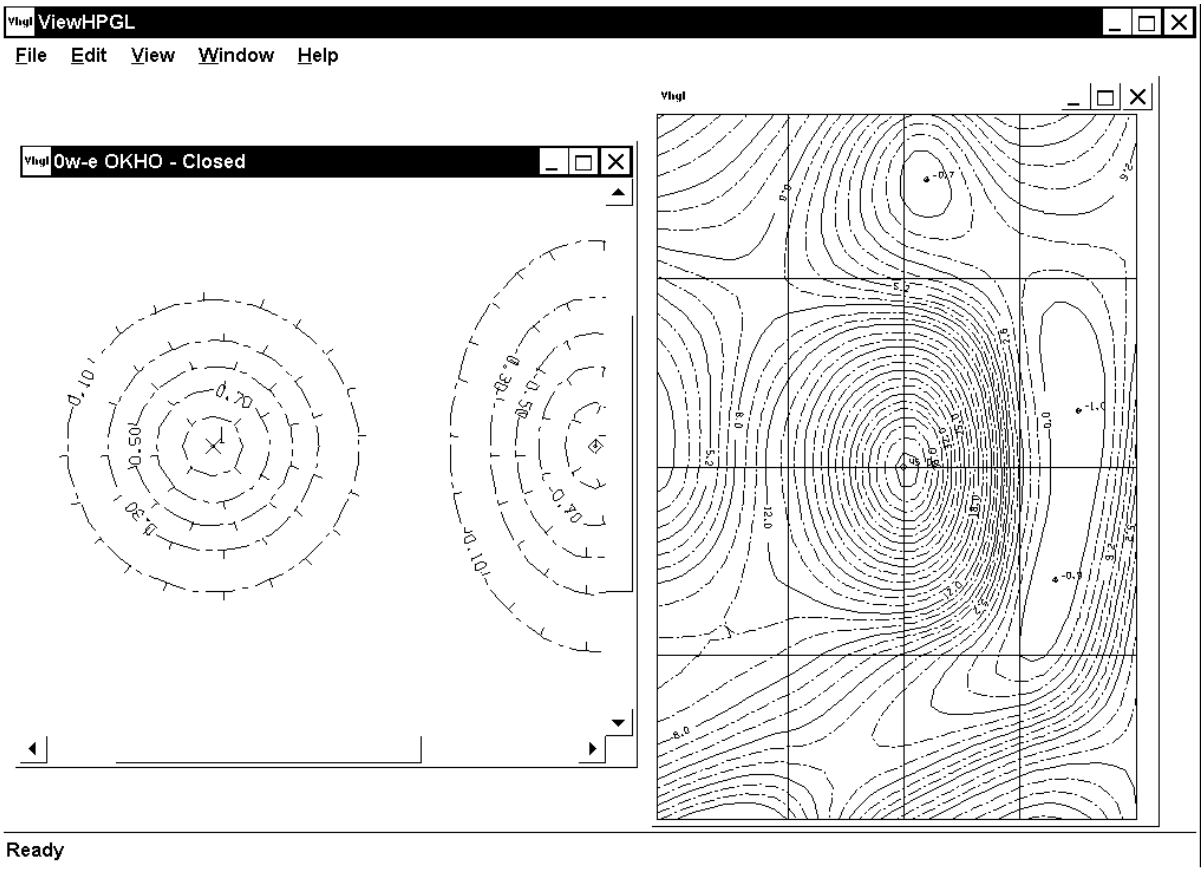
__declspec (dllimport) extern BOOL __stdcall ISHGLVIEW (HWND *hWnd*);

Функция проверяет является ли данное окно окном HPGL изображения.

Аргументы:	*hWnd*	– идентификатор (handle) тестируемого окна
Возвращаемое значение:	TRUE	– если указанный идентификатор принадлежит окну HPGL изображения.
	FALSE	– в противном случае.

Программа просмотра (Viewer).

Создана отдельная программа просмотра, которая позволяет просматривать файлы и после окончания работы с пакетом, поскольку по умолчанию они не уничтожаются и остаются на жестком диске в директории проекта. Пример работы приведен на рисунке.



Здесь, помимо возможностей передачи изображения в буфер обмена, масштабирования, подстраивания под размер, имеются возможности посмотреть, какая директория является текущей, задать просмотр одного конкретного файла или всех файлов из текущей директории, получить информацию о файле (полное имя, размер файла, дата создания и размер изображения в точках X на Y), а также сохранить текущее изображение в файле PS-формата. Сохранение изображения в файле PS-формата осуществляется специально написанной программой, о которой речь пойдет ниже в разделе “Запись графических файлов в формат Post Script”.

Описание меню.

Меню содержит команды работы с окнами просмотра HP-GL файлов.

File	–>	Open	– Выбрать HP-GL файл для просмотра через стандартное диалоговое окно Open
	–>	Open All	– Открыть для просмотра все HP-GL файлы из текущей директории

	–>	Export PostScript	– Сохранить текущее изображение в файле PS-формата
	–>	Close	– Закрыть активное окно просмотра HP-GL файла
	–>	Exit	– Закрыть все окна и завершить работу
Edit	–>	Copy .hgl file to the Clipboard	– Скопировать содержимое активного окна в буфер обмена
View	–>	Size To Fit	– Изменить масштаб и размеры изображения в соответствии с реальными размерами HP-GL файла
	–>	Zoom In	– Увеличить размер изображения
	–>	Zoom Out	– Уменьшить размер изображения
	–>	Info…	– Показать информацию о HP-GL файла активного окна просмотра : – полное имя, – размер файла, – дату создания и – размер изображения в точках X на Y.
	–>	Current Directory	– Показать текущую директорию
Window	–>	Cascade	– Расположить окна с перекрытием
	–>	Tile	– Расположить окна без перекрытия
	–>	Close All	– Закрыть все окна просмотра HP-GL файлов
Help	–>	About…	– Показать диалоговое окно с информацией о программе

Работа пакета ГРАФОР в Linux

Программа визуализации для пакета ГРАФОР на платформе UNIX написана с использованием функций графической библиотеки QT версии 2.3.1. Входом программы является файл в формате HP-GL. Каждому файлу соответствует об'ект класса Mwidget. Конструктор этого об'екта читает файл-результат, создает цепочку, каждый элемент которой – отрезок (цвет+координаты) в окне и инициализирует окно. Функция прорисовки окна берет информацию из цепочки.

Об'ект Mwidget создает окно, меню которого содержит пункты :

`"Close"` – закрывает окно,

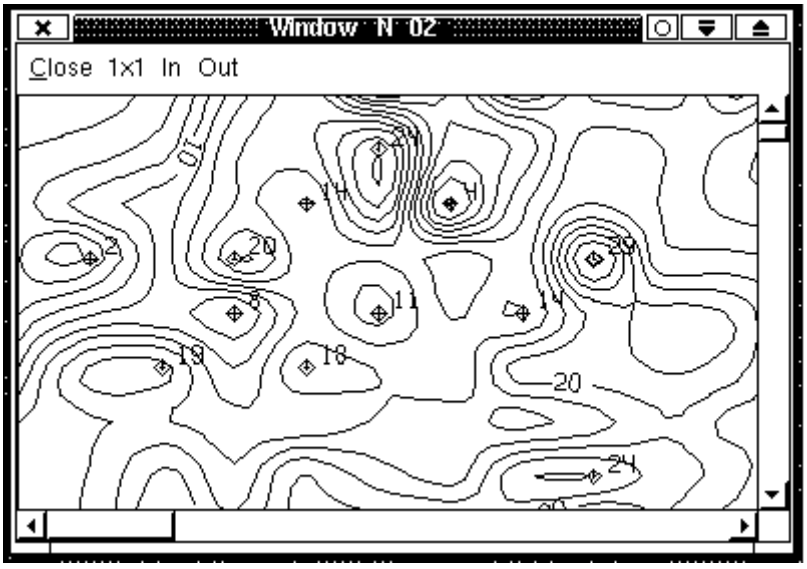
`"1x1"` – устанавливает масштаб 1x1,

"In" – увеличивает изображение в 2 раза,

"Out" – уменьшает изображение в 2 раза.

Если изображение не помещается в окне, то повляется соответствующая (X и/или У) полоса прокрутки. При закрытии последнего окна приложение заканчивает работу. Имена процедур в C/C++ не совпадают (добавлен символ подчеркивания в конце) с именами в Fortran'e.

Пример окна :



Для получения исполнимого модуля из ГРАФОРА и программы визуализации в LINUX предполагается использовать 3 модуля :

1) Модуль, написанный на Fortran - fortran.f.

2) Модуль, написанный на C – sig.c, нужен для связи с модулем main.cpp.

3) Модуль, написанный на C++ – main.cpp, реализует визуализацию результатов, полученных в fortran.f.

Взаимодействие модулей должно осуществлятся так :

1. Запускается функция main() модуля main.cpp. Она назначает процедуру signal1() обработчиком сигнала SIG_USR1, подготавливает среду для создания окон приложения и вызывает главную процедуру модуля grafor.f, передавая ей без изменений все полученные при запуске параметры.

2. В модуле grafor.f после создания файла-результата для его отображения вызывается процедура dowin1(),

3. Процедура dowin1() модуля sig.c изменяет глобальную переменную num_win (в данном случае в 1) и посылает приложению (то есть себе) сигнал SIG_USR1.

4. Процедура signal1() модуля main.cpp создает динамический объект класса MWidget.

Проведено предварительное тестирование предложенного способа - роль grafor.f выполнял модуль callwin.f.

```
  Исходные тексты :
/********** begin callwin.f ***************/
         subroutine fortran
         integer argc
         character(*) argv
         call dowin1
         call dowin2
         end
/********** end callwin.f ***************/

/************* begin sig.c *************/
#include <signal.h>
char num_win;
void dowin1_()
{
  num_win=1; kill( getpid(), SIGUSR1);
}
void dowin2_()
{
  num_win=2; kill( getpid(), SIGUSR1);
}
/************* end sig.c **************/
// фрагмент  main.cpp   Вариант для сборки вместе с callwin.f , sig.c.
/*
   Описание класса VWidget
*/

#define TEST
#ifdef TEST
extern char num_win;
extern "C" void fortran_(int argc, char **argv);

void signal1(int sig)
{  MWidget *m=new MWidget(0,0, num_win, 300, 200 ); }
#endif

int main( int argc, char **argv )
{
#ifdef TEST
    signal( SIGUSR1, signal1 );
#endif
    QApplication a( argc, argv );
    QObject::connect( qApp, SIGNAL( lastWindowClosed() ), qApp, SLOT(
quit() ) );

#ifdef TEST
    fortran_( argc, argv);
#else
    MWidget *m1=new MWidget(0,0, 1, 300, 200 );
    MWidget *m2=new MWidget(0,0, 2, 300, 200 );
//   . . .
//   + another windows
#endif

    return a.exec();
}

//   main.cpp
```

Формат Post Script

Наряду с файлами на языке HP-GL удобно получать файлы на языке PostScript, который является межплатформенным и также имеет текстовый формат. Кратко опишем особенности этого языка.

Файл EPS состоит из двух частей – Prolog и Script, каждая из которых может включать в себя управляющие операторы EPS. Содержащиеся в теле файла управляющие операторы EPS начинаются, как и операторы языка PostScript, с символа процента – “%”. Таким образом, EPS-файлы могут обрабатываться как обычным транслятором с языка PostScript, который такие опреаторы будет игнорировать, так и более сложными программами обработки, которыми операторы EPS будут интерпретироваться. Признаком опратора EPS, позволяющего отличить его от комментария, служит второй символ оператора, следующий за “%”. Управляющие операторы начинаются символами “%!” или ”%%”, за которыми следует ASCII-строка. Первый оператор выглядит так (W, X, Y, Z – конкретные номера версий и подверсий) :

%!PS-Adobe-W.X EPSF-Y.Z

При записи операторов языка PostScript используется обратная польская кодировка. Приведем примеры операторов, которые используются нами при “переводе” файлов из формата HP-GL в PostScript:

Оператор: lineto
Формат: x y lineto
Функция: Присоединяет к текущей области вывода отрезок прямой, ограниченной точкой с координатами *(x,y)* и текущей точкой.

Оператор: moveto
Формат: x y moveto
Функция: Устанавливает в качестве текущей точку с координатами *(x,y)*.

Запись графических файлов в формат Post Script

Реализована подпрограмма, осуществляющая чтение файлов (записанных при работе ГРАФОРа) в формате HP-GL и запись в PostScript соответствующих операторов. Программа реализована на ФОРТРАНе и работает как в WINDOWS, так и в LINUXе. Возможны передача файла, полученного в одной системе, в другую, и просмотр его (в LINUX есть “штатный просмотрщик” PS-файлов).

Реализована и возможность использования этой подпрограммы непосредственно в драйвере ГРАФОРа, однако, как нам кажется, её использование уже “избыточно”, поскольку пользователь, получив ряд изображений, может выбрать только те, которые ему необходимы, и осуществить конвертацию только нужных ему изображений.

Получение изображения двумерного поля параметров в формате PostScript.

Приведем текст программы, записывающей изображение двумерного поля параметров (в данном примере поле задается по аналитической формуле), в файл формата PostScript. Просмотреть получающийся файл (в системе WINDOWS) можно широко распространенной программой Gsview, а в операционной системе Linux – имеющейся в ней штатной программой просмотра.

```
open (3,file='test_ps.ps',status='unknown')
write (3,*)'3 dup scale'
write (3,*)'0 setlinewidth'
write (3,*)'/L{lineto}def /M{moveto}def'
write (3,*)'/C{sethsbcolor}def /F{fill}def /S{stroke}def'
do j=1,100
do k=1,100
xcol=2.0+( cos(float(k)) + sin(float(j)) )
write (3,'(2I4,2H M,3(2I4,2H L))')j-1,k-1,j,k-1,j,k,j-1,k
write (3,'(F6.3,10H 1 1 C F S)') xcol
enddo
enddo
write (3,*) 'showpage'
write (3,*) 'quit'
close (3)
end
```