УДК 004.427

DOI: 10.21779/2542-0321-2024-39-1-30–36

А.М. Магомедов, Н.Ш. Раджабова

Интерактивное построение графика функции[1]

Дагестанский государственный университет; Россия, 367000, г. Махачкала, ул. М. Гаджиева, 43а; magomedtagir1@yandex.ru, naimasha@gmail.com

Аннотация. В учебниках по Delphi, как правило, недостаточно внимания обращают на такие вопросы, как применение директив компилятора или, например, запуск других приложений из авторской программы. Авторы учебников, видимо, исходят из того, что начинающему программисту столь продвинутые средства не понадобятся, а специалист вряд ли обратится к учебнику.

Однако именно использование упомянутых средств позволяет даже начинающему программисту создавать востребованные в прикладных областях приложения.

В статье дано подробное руководство по разработке программы построения графика функции, аналитическое представление *f(x)* которой не встроено в текст проекта и может быть введено в интерактивном режиме.

Суть решения заключается в следующем. На первом шаге создается вспомогательный проект вывода графика функции f, заключительный оператор реализации которой предписывает присвоить имени функции содержимое некоторого текстового файла (с помощью директивы вставки).

Основной проект создается на втором шаге. Вначале пользователю предлагается ввести в интерактивном режиме аналитическое выражение для функции; основной проект немедленно записывает его в упомянутый выше текстовый файл. Затем вызывается системная функция запуска внешнего приложения. Роль этого внешнего приложения играет компилятор командной строки, которому в качестве параметра передается имя используемого файла (перед началом компиляции директива компилятора обеспечивает восстановление целостности текста вспомогательного проекта). В результате создается исполняемый файл вспомогательного проекта.

В заключение в том же основном проекте вторично вызывается системная функция запуска внешнего приложения, на этот раз – для запуска созданного исполняемого файла вспомогательного проекта.

Ключевые слова: программирование, Delphi, график функции, параметр, файл.

Введение

Статья содержит материал, который может быть полезен для тех, кто хочет быстро научиться рисовать графики функций любой сложности средствами Delphi. Обычно в таких программах предусматривают ввод некоторой числовой информации в интерактивном режиме (например, границ изменения аргумента функции). В частности, данная заметка позволит включить в список вводимых в интерактивном режиме данных аналитическое выражение самой функции.

Отметим, что учебная литература по Delphi весьма обширна (см., например, [1–14]). Для углубленного изучения мира компьютерной графики рекомендуем книги [15–20].

[1] Работа выполнена при поддержке Отдела математики и информатики ДФИЦ РАН.

1. Построение графика функции

1.1. Схема построения графика функции одной переменной

График заданной функции $y = f(x)$ на странице клетчатой тетради обычно рисуют так: разбивают область определения $[x_1; x_2]$ на n частей, вычисляют значения функции в точках $x = x_1, x_1 + \Delta x$, ..., x_2, где $\Delta x = (x_2 - x_1)/n$, и слева направо соединяют полученные точки (x, y). При этом умение находить область значений $[y', y'']$ способствует успешному выбору ограничивающего прямоугольника и единичного отрезка по оси OY. Единичный отрезок по оси OX обычно принимают равным стороне клетки.

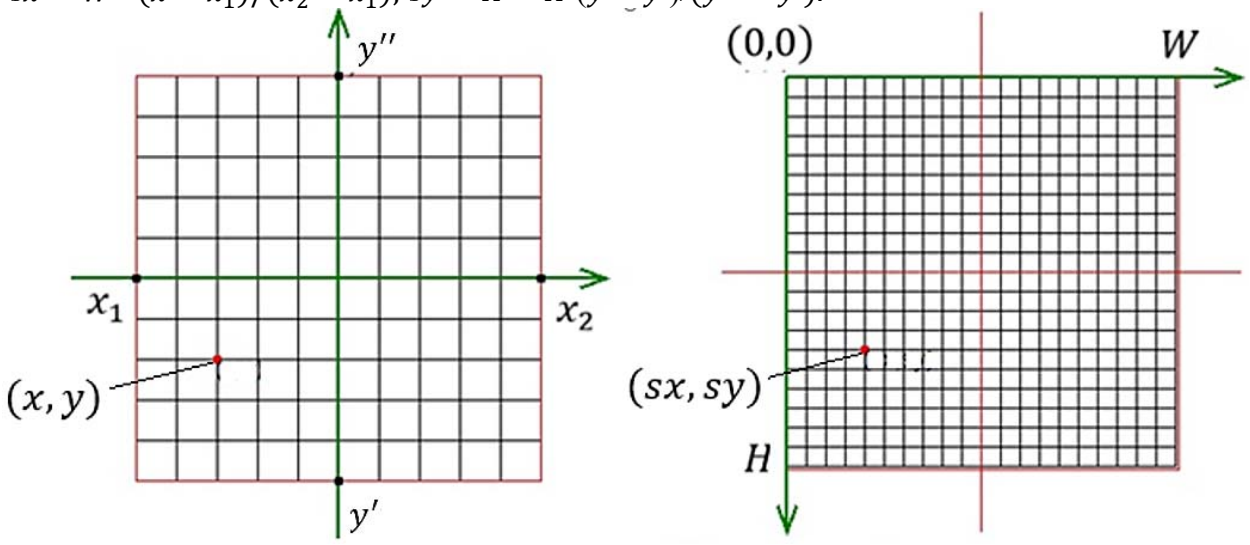
Идея переноса графика в окно формы заимствована из [13] и заключается в преобразовании «бумажных» координат точек (x, y) графика и осей координат в оконные координаты, которые мы будем обозначать через sx и sy соответственно.

1.2. Преобразование координат

Начало оконной системы координат располагается в верхнем левом углу окна, ось игреков направлена вниз, координаты точек – целые неотрицательные числа. На рис. 1*б* через W и H обозначены соответственно ширина и высота окна.

При движении точки (x, y) *в бумажном варианте от левого до правого края ограничивающего прямоугольника доля пройденного расстояния* $(x - x_1)/(x_2 - x_1)$ *равна* sx/W *– доле расстояния, пройденного точкой* (sx, sy) *в окне*. Аналогично $(y - y')/(y'' - y') = (H - sy)/H$. Отсюда:

$sx = W \cdot (x - x_1)/(x_2 - x_1)$; $sy = H - H\ (y - y')/(y'' - y')$.



1а 1б

Рис. 1: а) – бумажная система координат; б) оконная система координат

1.3. Вывод графика

Для вывода графика функции $y = f(x)$ разобьем отрезок $[x_1; x_2]$ на n частей длины $\Delta x = (x_2 - x_1)/n$ и установим инструмент для рисования (*ручку*) в самую левую точку графика функции:

$$x := x_1;\ y := f(x);\ \ moveTo\ \big(SX(x), SY(y)\big).$$

Увеличивая в n раз значение x на Δx, проводим каждый раз линию из предыдущей точки в точку $(x, f(x))$:

$for\ i := 1\ to\ n\ do\ begin$
$x := x + deltaX;\ y := f(x);\ LineTo(Sx(x), Sy(y));$
$end.$

Следующие стандартные действия не представляют труда: – установка цвета фона и точек графика; – вывод стрелок на осях координат; – нанесение на оси насечек и цифр. Легко также реализовать анимированный вывод графика и изменить толщину и стиль линий или использовать процедурный тип для вывода нескольких графиков на одном рисунке. Определенные трудности могут вызвать следующие две задачи: 1) вывод графика функции, аналитическое выражение для которой вводится в интерактивном режиме, а не задается в модуле проекта; 2) вывод графика функции двух переменных. Решение первой задачи см. в §3; о выводе графика функции двух переменных можно прочитать [13, с. 444–453].

2. Подробное решение

2.1. Объявление полей и методов класса TForm1

В разделе Private класса TForm1 объявим переменные:

```
W, H: integer; deltaX, x, y: real;
```

и перечислим заголовки методов класса:

```
procedure MyGraphic;
function SX(x: real): integer;
function SY(y: real): integer;
function f(x: real): real;
```

2.2. Реализация методов SX и SY преобразования координат

В части implementation модуля Unit1 объявим константы

```
x1=-5.0; x2=5.0; y1=-5; y2=5; n=200;
```

затем приведем реализации методов преобразования и укажем аналитический вид функции $f(x)$, график которой мы хотим вывести на холст формы:

```
function TForm1.SX(x: real): integer;
begin  SX:=round ((x-x1)/(x2-x1)*W); end;
function TForm1.SY(y: real): integer;
begin  SY:=round (H-(y-y1)/(y2-y1)*H); end;
function TForm1.f(x: real): real;
begin  f:= Cos(3*x)*exp(-0.3*x); end;
```

2.3. Заключительные действия

Осталось привести реализацию основного метода MyGraphic и вызвать метод, скажем, из обработчика события onPaint для формы.

```
procedure TForm1.MyGraphic;
var i: integer; x,y: real;
 begin
   with canvas do
   begin //позиционируем ручку в точке (x1, f(x1))
    x:=x1; y:=f(x);  moveTo (SX(x), SY(y));
//увеличивая x на Δx, проводим линию в точку (x, f(x))
    for i := 1 to n do  begin  x:=x+deltaX; y:=f(x);  LineTo (SX(x), SY(y));  end;
   end;
end;
```

В обработчике события onPaint для формы вычислим значения $W, H, \Delta x$, проведем оси координат и вызовем метод MyGraphic:

```
procedure TForm1.FormPaint(Sender: TObject);
begin
 W:=clientwidth; H:=clientheight; deltaX:= (x2-x1)/n;
 with canvas do
   begin
     x:=x1; y:=0; moveTo (SX(x1), SY(y)); LineTo (SX(x2), SY(y));
     x:=0; y:=y1; moveTo (SX(x), SY(y)); LineTo (SX(x), SY(y2));
    end;
 MyGraphic;
end;
```

3. Ввод выражения для функции в интерактивном режиме

3.1. Перемещение выражения для $f(x)$

В выше приведенной реализации метода $f(x)$ выражение в правой части оператора присваивания жестко встроено в текст модуля Unit1.pas проекта project1.dpr:

```
f:= cos(3*x)*exp(–0.3*x)                                    (1)
```

Нельзя ли организовать задание текста выражения для функции по приглашению программы?

В папке нашего проекта создадим (например, с помощью Блокнота) текстовый файл ‘1.txt’ с текстом

```
cos(3*x)*exp(-0.3*x),
```

одновременно заменим (1) на директиву компилятора

```
f:={$I 1.txt}
```

Скомпилируем и выполним программу. Как мы увидим, подобное «отчуждение» текста выражения для функции не вызвало принципиальных затруднений, т. к. благодаря директиве компилятора восстанавливается целостность текста.

Выше мы вручную выполнили ряд действий:

– создали в текущей папке файл и записали в него аналитическое выражение для функции, график которой требуется построить (первое действие);

– затем скомпилировали (второе действие) project1.dpr, зная, что компилятор, следуя директиве, восстановит целостность текста модуля;

– запустили (третье действие) созданный в результате компиляции исполнимый файл projec1.exe.

В следующем разделе выполнение всех этих трех действий поручим консольному проекту project2.dpr, который создадим в той же папке.

3.2. Проект project2

```
program Project2;
{$APPTYPE CONSOLE}
uses
 SysUtils, Windows, Dialogs;
var
 g: Text;
 s: string;
begin
//введем аналитическое выражение для f(x)
```

```
s:=inputBox ('Рисуем график.',
'Введите выражение для функции:', 'sin(x)');
//запишем его в файл 1.txt
 AssignFile (g,'1.txt');
 rewrite(g);
 WriteLn(g,s);
 CloseFile(g);
//скомпилируем project1.dpr с помощью компилятора командной строки
 winexec ('dcc32.exe project1.dpr',0);
 sleep(1000);
//выполним его
 winexec ('project1.exe',1);
end.
```

Заключение

Для построения графика функции одной переменной запускаем project2.exe, который предложит ввести выражение для функции (рис. 2).

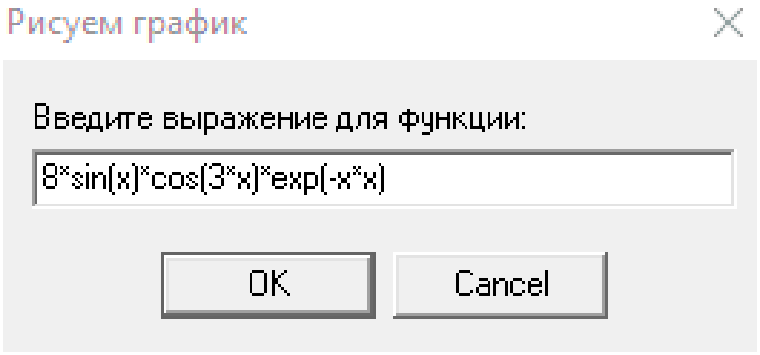


Рис. 2. Вводим выражение для функции f(x)

В окошко, где мигает курсор, введем выражение любой сложности, составленное по правилам Delphi. После тщательной проверки нажмем для подтверждения кнопку OK и увидим график (рис. 3).

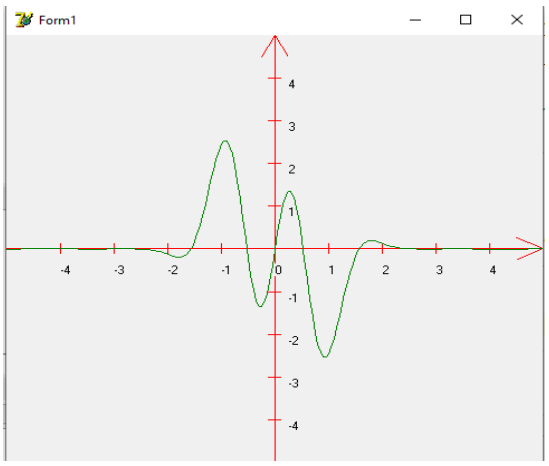


Рис. 3. График функции, выражение для которой задано интерактивно

Литература

1. Delphi C/S 2. Русскоязычная документация; Borland Press. – М., 2015. – 321 с.
2. *Архангельский А.Я.* Программирование в Delphi 6; Бином. – М., 2018. – 258 с.
3. *Белов В.В., Чистякова В.И*. Программирование в Delphi: процедурное, объектно-ориентированное, визуальное; учебное пособие для вузов. – М.: РИС, 2014. – 240 с.
4. *Бобровский С.* Delphi 7. Учебный курс. – СПб.: Питер, 2018. – 736 с.
5. *Григорьев А.Б.* О чем не пишут в книгах по Delphi. – СПб.: БХВ-Петербург, 2016. – 576 с.
6. *Дарахвелидзе П.Г., Марков Е.П.* Delphi 2005 для Win32 (наиболее полное руководство). – СПб.: БХВ-Петербург, 2018. – 234 с.
7. *Культин Н*. Основы программирования в Delphi 7. – СПб.: БХВ-Петербург, 2014. – 608 с.
8. *Марков Е.П., Никифоров В.В*. Delphi 2005 для .NET. – СПб.: БХВ-Петербург, 2017. – 896 с.
9. *Нагаева И.А*., *Кузнецов И.А*. Программирование: Delphi; учебное пособие для среднего профессионального образования / под ред. И.А. Нагаевой. – М.: Юрайт, 2023. – 302 с.
10. *Осипов Д.* Delphi. Профессиональное программирование. – СПб.: Символ-плюс, 2015. – 1056 с.
11. *Сван Т.* Секреты 32-разрядного программирования в Delphi: Диалектика. – М., 2015. – 480 с.
12. *Сухарев М.В.* Основы Delphi. Профессиональный подход: Наука и техника. – М., 2018. – 600 с.
13. *Тюкачев Н.* Программирование графики в Delphi. – СПб.: БХВ-Петербург 2021. – 784 с.
14. *Эйдлина Г.М., Милорадов К.А.* Delphi: программирование в примерах и задачах. Практикум: учебное пособие. – М.: Риор, 2018. – 76 с.
15. *Дегтярев В.М., Затыльникова В.П.* Инженерная и компьютерная графика: учебник. – М., 2010. – 238 с.
16. *Порев В.Н.* Компьютерная графика. – СПб., 2021. – 428 с.
17. *Евченко А.И.* OpenGL и DirectX: программирование графики. – СПб., 2006. – 349 с.
18. *Коцюбинский А.О.* Компьютерная графика: практ. пособие. – М., 2001. – 750 с.
19. *Логиновский А.Н.* Инженерная 3D-компьютерная графика: учебное пособие для бакалавров. – М.: Юрайт, 2013. – 464 с.
20. *Миронов Д.Ф.* Компьютерная графика в дизайне: учебник. – СПб.: БХВ-Петербург, 2021. – 560 с.

Поступила в редакцию 1 марта 2024 г.
Принята 13 марта 2024 г.

UDC 004.427

DOI: 10.21779/2542-0321-2024-39-1-30–36

Interactive Plotting of a Function

A.M. Magomedov, N.Sh. Radjabova

Dagestan State University; Russia, 367000, Makhachkala, M. Gadzhiev st., 43a; magomedtagir1@yandex.ru, naimasha@gmail.com

Abstract. Textbooks on the Delphi language, as a rule, pay little attention to such issues as the use of compiler directives or, for example, launching other applications from the author's program. **Abstract.** Textbooks on the Delphi language, as a rule, pay little attention to issues such as the use of compiler directives or, for example, launching other applications from the author's program. The authors of the textbooks seem to assume that a novice programmer will not need such advanced tools, and an experienced programmer is unlikely to turn to the textbook.

However, it is the use of these tools that allows even a novice programmer to create applications that are in demand in applied fields.

This note provides detailed guidance on the development of a program for plotting a function, the analytical representation of which f(x) is not embedded in the text of the project and can be entered interactively.

The essence of the solution is as follows. As the first step, an auxiliary project is created for the output of the graph of the function f, the final implementation operator of which prescribes assigning the contents of a certain text file to the name of the function (using the insertion directive).

The main project is created in the second step. First, the user is prompted to interactively enter an analytical expression for the function; the main project immediately writes it to the text file mentioned above. Then the system launch function of the external application is called. The role of this external application is played by the command-line compiler, to which the name of the file used is passed as a parameter (before starting compilation, the compiler directive ensures the restoration of the integrity of the text of the auxiliary project). As a result, an executable file of the auxiliary project is created.

Finally, in the same main project, the system launch function of the external application is called a second time, this time – to launch the executable file of the auxiliary project.

Keywords: programming, Delphi, function graph, parameter, file.

Received 1 March, 2024
Accepted 13 March, 2024