Дебелов Виктор Алексеевич

Векторные поля на Java

Задача Vect

Построить приложение, рисующее векторное поле по двум заданным однозначным функциям 2-х переменных.

1. Цветное векторное поле. Даны две однозначные функции

fx(x,y), (x,y) є D = [a,b]*[c,d],

fy(x,y), (x,y) є D = [a,b]*[c,d].

Эти функции определяют поле скоростей (например, ветра или воды) на области пределения **D**. Другими словами в каждой точке **(x,y) є D** мы имеем вектор

(fx(x,y), fy(x,y)).

Пусть **L(x,y) –** длина этого вектора.

Пусть даны **n** чисел:

l1 < l2 < … < ln,

и **n+1** цвет:

c0, c1, c2, …, cn.

Будем говорить, что значению **z** соответствует цвет **c0**, если **L(x,y) < l1**, иначе значению L соответствует цвет **ci (i = 1..n)**, где **i** определяется как

i = min(j) {L >= lj}.

В зависимости от длины вектора он рисуется тем или иным цветом.

На клиентской области окна приложения выбираем прямоугольник **P** с углами **(u0,v0)** и **(u1,v1)** в экранных координатах (пикселях), ось **V** – *снизу вверх*. Должно сохраняться отношение:

$$\frac{u_1 - u_0}{b - a} = \frac{v_1 - v_0}{d - c}, \quad \text{(aspect ratio)}$$

т.е. сохраняем соотношение размеров между исходной областью определения **D** и её изображением **P**. Это означает, что на области окна, выделенной под **P**, могут оказаться пустые вертикальные или горизонтальные полосы.

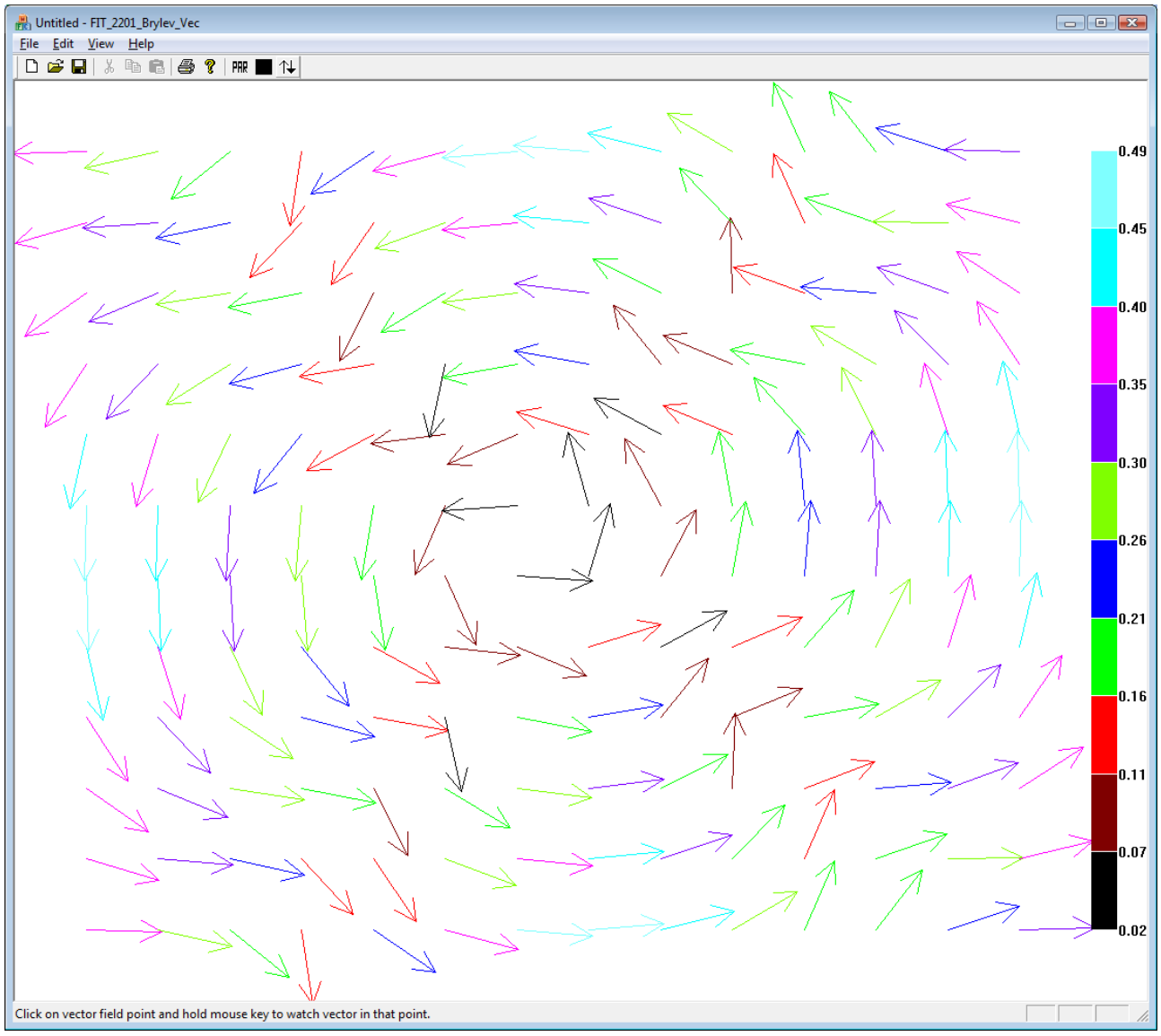
Рассмотрим довольно простое отображение **D** на **P**, когда точка **(a,c)** переходит в пиксель **(u0,v0)**, а точка **(b,d)** в **(u1,v1)**. Очевидное обратное преобразование позволяет по центру каждого пикселя **(u,v)** из **P** получать точку **(x,y)** из **D**, и, таким образом, получать соответствующее пикселю значение **f(x,y)**.

На области **D** наносим равномерную сетку (в уме) **MxN клеток**. Таким образом, мы получаем те точки **{(xi,yj), i=0..M, j=0..N}**, в которых мы должны вычислить векторы нашего поля и нарисовать.

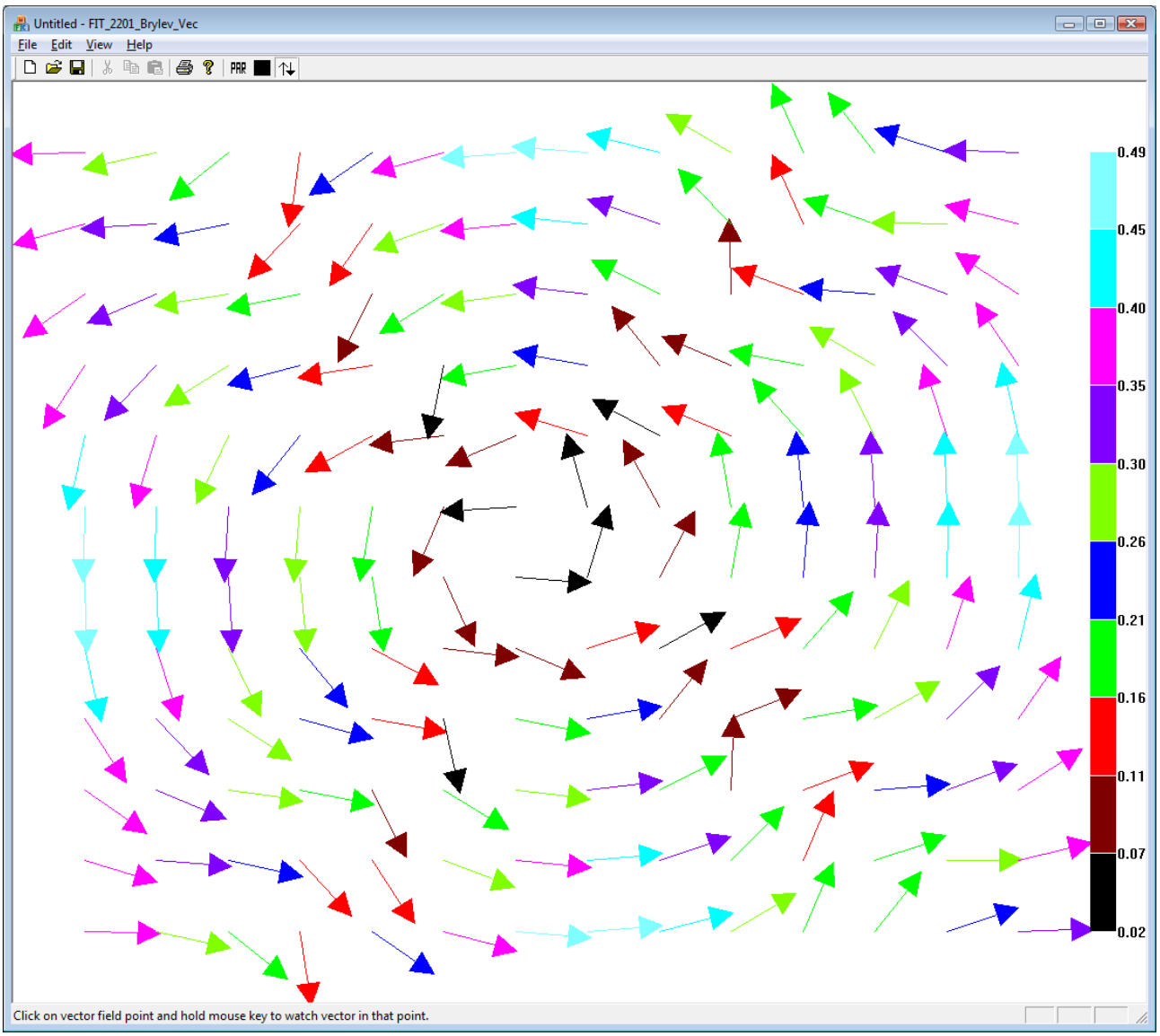
Выбор длины вектора. Размер вектора не должен превышать величины **C0*S,** где **C0** – параметр в пределах от 0.5 до 3.0, а **S** – величина диагонали клетки.

Изображение вектора. Рисовать вектор в виде стрелки. Обязательный режим: стрелка в виде двух «усиков». Дополнительные (необязательные) режимы отображения зависят от фантазии исполнителя.

Справа от прямоугольника на свободном месте окна приложения рисуется вертикальная *легенда*, показывающая соответствие цвета величине скорости. Все вектора в данном режиме (цветном) изображения рисуются одинаковой длины. Направление определяется значениями заданных функций.

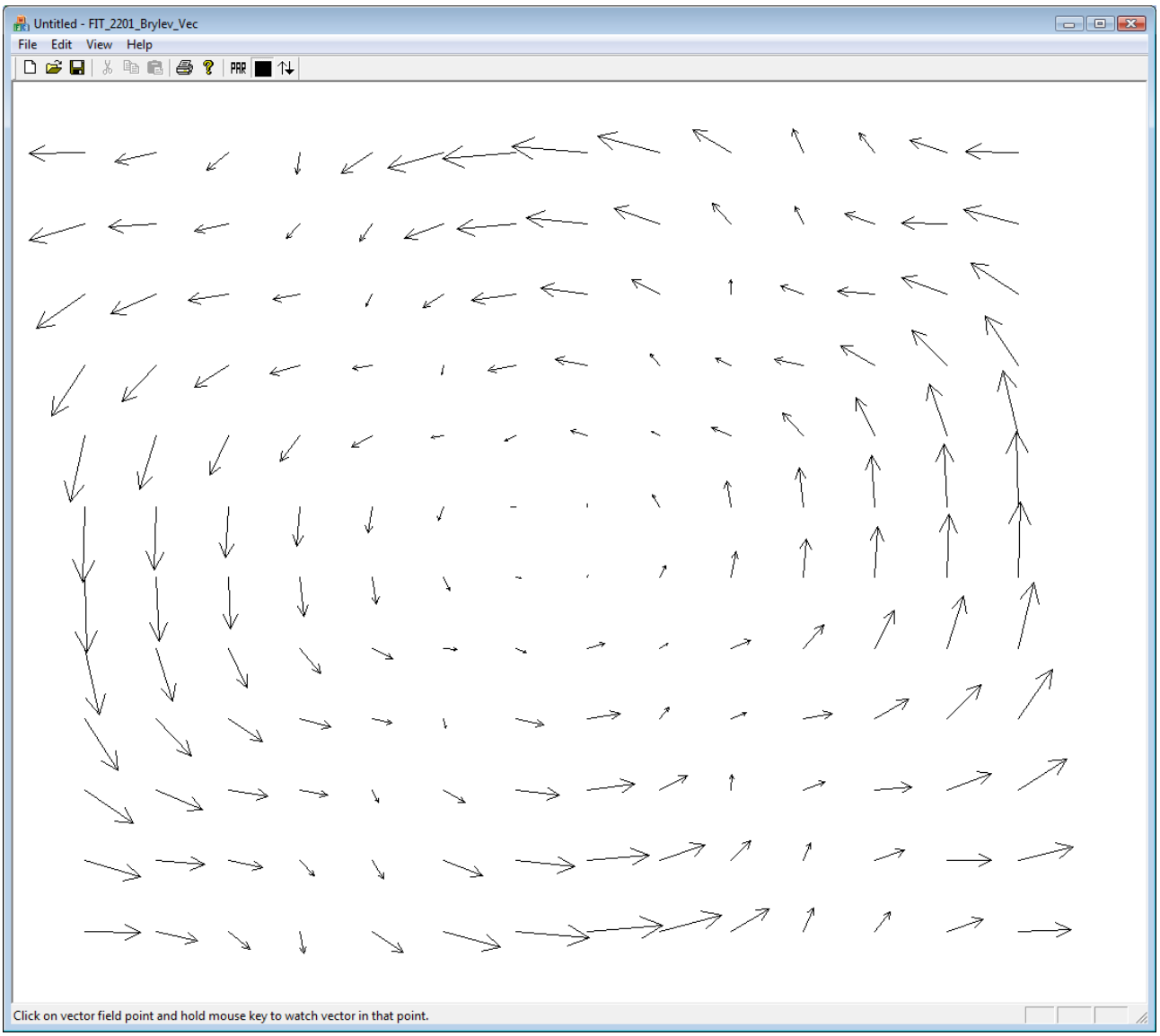


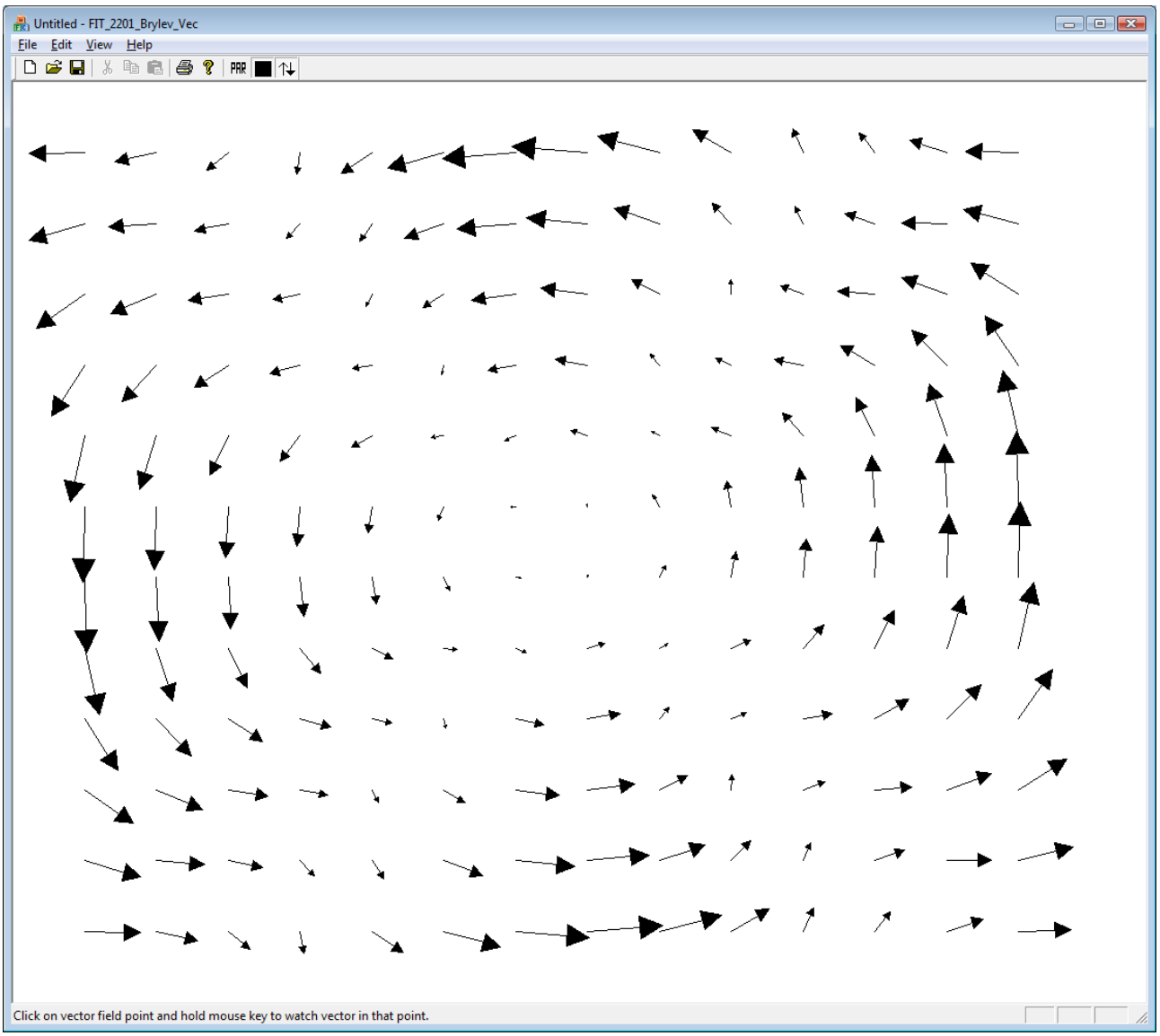
Можно ввести дополнительный вид стрелок, как здесь:



2. Серое векторное поле.

Программа должна работать в 2-х режимах, которые переключаются специальной кнопкой. Первый режим "цветного поля" описан выше. В режиме серого векторного поля все вектора рисуются одним цветом. НО длины векторов пропорциональны, т.е., на рисунке выводятся вектора длины пропорциональной **L(x,y)**. При этом размер максимального вектора не должен превышать величины **C0*S**. Для очень малых длин (задать свою константу) рисуется только отрезок без наконечника стрелки. Может оказаться, что получится вырожденный отрезок – точка – для нулевого вектора. Для упрощения цветную легенду можно оставлять и на данном виде. Можно и убирать, т.е. **P** будет разным для цветного и серого режимов. (Больше работы).


Untitled - FIT_2201_Brylev_Vec
File Edit View Help
Click on vector field point and hold mouse key to watch vector in that point.



Параметры задачи

- **M, N**.
- **fx(x,y), fy(x,y)** – аналитически заданные функции. Выбор конкретной функции - за исполнителем.
- **a, b, c, d** – вводятся из файла, это **a0, b0, c0, d0**. **a, b, c, d** можно менять в диалоге, но не больше, чем **a0, b0, c0, d0,** которые будут введены из файла.
- растр модели – прямоугольник **P** с углами **(u0,v0)** и **(u1,v1)** в экранных координатах – вводится из файла.
- **C0** – вводится из файла;
- **n** – вводится из файла;
- **c0..cn** – вводятся из файла в виде 255 170 24.

- **l1..ln** – для определения этих значений: подсчитываются минимум и максимум длины вектора, в полученном интервале равномерно распределяются остальные значения.

Также все эти параметры можно задавать/изменять, используя диалоги:

Полученную карту векторного поля вместе с легендой вписать в прямоугольник **P**. Не забыть поля (бордюры) между разными зонами (с границей окна, между легендой и векторным полем) – пикселей 5–10.

При движении мыши по полю в нижней строке окна писать значения истинных (не растровых) значений x, y, fx, fy.

Замечание: легенда состоит не только из палитры используемых цветов, но и цифровых значений уровней, которые подписываются СБОКУ, а не ПО легенде.

Дополнительно для желающих

Вектора можно выводить не для всех узлов сетки, а например, выбирать их в шахматном порядке, т.е. можно завести такой режим.

При движении мыши по полю под легендой можно отображать вектор, соответствующий точке под мышью. Можно его отображать прямо в месте курсора мыши.

- Выбор **a, b, c, d** можно делать неявно, используя резиновый прямоугольник. Изначально **a, b, c, d** совпадают с **a0, b0, c0, d0.**

Другое разумное по желанию.

Файл.

- **a0 b0 c0 d0 //** начальные, дать возможность менять в программе
- **C0**
- **N** // начальное значение числа градаций цвета
- **c0..cn** // вводятся из файла в виде 255 170 24. Одно значение цвета на строку файла
- **M N** // начальные, дать возможность менять в программе
- **cs** // цвет сетки
- EOФ

Замечания по реализации

1. Выбор **P** на области окна, которое имеет размеры АхБ:
    a. Рассчитать ширину в пикселях для легенды – Шл
    b. Задать величину бордюра для легенды – Рб
    c. Под векторное поле выделяется по горизонтали А1 = (А – 2*Рб – Шл), т.е. учли все бордюры (отступы)
    d. У нас сетка значений функций MxN или (M – 1) x (N – 1) клеток. Для того чтобы выходящие вовне области **D** векторы всё-таки были видны, зададим обрамление шириной в 1 клетку. Таким образом, у нас появляется виртуальная область Д. Это **D**, расширенная на 1 клетку во всех направлениях: 2 по вертикали и 2 по горизонтали. Таким образом, у нас исходная область по

горизонтали как бы имеет размер $rx = \frac{M+1}{M-1}(b-a)$, а по вертикали – $ry = \frac{N+1}{N-1}(d-c)$.

e. Рассчитать реальные размеры в пикселях, отводимые под отображение виртуального прямоугольника [rx,ry]. Полный размер = А1хБ. Чтобы сохранить (aspect ratio) нам придется либо по вертикали, либо по горизонтали оставить пустые полосы, это В ОБЩЕМ СЛУЧАЕ! Рисунок должен быть отцентрирован в области А1хБ пикселей. Таким образом мы получили прямоугольник **P**.

f. Внимание! Пока реализуйте к-л выбор прямоугольника **P**. А при очной встрече разберем этот расчет подробно. Иначе мы этим вопросом завалим форум. Пока одно: P и D и aspect ratio.

2. Должна быть кнопка, вкл/выкл отображения сетки MxN, обрамляющие клетки не показываются. Не путайте понятия сетки и клетки. Сетку отображать пунктирной линией и рисовать под полем, т.е. первой.
3. При движении мыши значения в статус баре отображаются только в том случае, когда мышь «находится» в области **D**, т.е. в прямоугольнике **P**.
4. **Дополнительно** можно рисовать соответствующую стрелку: а) прямо в точке нахождения курсора мыши; б) в малом окошке под легендой.
5. Поиск минимума и максимума длины вектора делается по значениям в узлах сетки.
6. Нулевые вектора всегда рисуются в виде точки
7. Легенда должна иметь вид
8. Кнопки ещё раз (минимальный набор), кроме стандартных: чтения/запись/new/About/Exit …
    a. Вкл/Выкл сетки
    b. Серое/цветное поле
    c. Изменение параметров C0, M, N, a, b, c, d, n. n не более начального значения (из файла), от 4 до 20. M, N от 4 до 50. C0 от 0.3 до 3.0
    d. a, b, c, d **дополнительно** можно менять резиновым прямоугольником
    e. **дополнительно** смена типа стрелки
    f. **дополнительно** задавать любое n до 20, задавать новые цвета и изменять старые. В этом случае легенда пересчитывается.
    g. Можно свой набор, но с аналогичной совокупной функциональнстью.
9. При записи в файл пишутся текущие значения переменных
10. При изменении **P** (ресайз окна) изображение пересчитывать
11. Легенда рассчитывается один раз, после чтения файла
12. **Изменяется стратегия работы с файловой системой**. В предыдущих задачах файлы **всегда** открывались в дир FIT_...._Data. Начиная с данной задачи:

a. При чтении или записи первый раз после открытия программы открывается также дир FIT_...._Data. При всех последующих чтениях/записях программа выходит в ту дир, в которой реально были произведены чтение или запись.

13. Срок: 12-00 11 апреля 2011г.

Примеры приложений, как обычно, не являются решением данной постановки. Но позволяют выяснить ряд моментов реализации.

Это стартовый вариант задания. Мы его должны обсудить, и оставшиеся детали утвердить.