Введение в LaTeX

Занятие 4

Даниил Дрябин, Алексей Ребриков

Студсовет ФПМИ

весна 2022

Счетчики в теоремах

Вложенные счетчики

Вспомним, как показать, что счетчики одного типа теорем нумеруются на один уровень глубже, чем некоторый другой счетчик (да, это не обязан быть другой тип теорем!):

```
\theoremstyle{plain}
\newtheorem{theorem}{Теорема}[section]
\newtheorem{corollary}{Следствие}[theorem]
\newtheorem*{definition}{Определение}

\section{Первый раздел}
\begin{theorem}Текст.\end{theorem}
\begin{theorem}Еще текст.\end{theorem}
\begin{proof}Тривиально.\end{proof}
\begin{corollary}И еще текст.\end{corollary}
\begin{definition}Что-то новое.\end{definition}
```

Вложенные счетчики

Код с предыдущего слайда дает следующий результат:

1 Первый раздел

Теорема 1.1. *Текст.*

Теорема 1.2. *Еще текст.*

Доказательство. Тривиально. □

Следствие 1.2.1. *И еще текст.*

Определение. *Что-то новое.*

Общие счетчики

Разные типы теорем могут иметь общий счетчик, инкрементирующийся при объявлении теоремы любого из этих типов.

```
\newcounter{mycount}

\theoremstyle{plain}
\newtheorem{proposition}[mycount]{Утверждение}
\newtheorem{lemma}[mycount]{Лемма}

\begin{proposition}Текст.\end{proposition}
\begin{lemma}Второй текст.\end{lemma}
\begin{proposition}Третий текст.\end{proposition}
\stepcounter{mycount}
\begin{lemma}Четвертый текст?\end{lemma}
```

Общие счетчики

Код с предыдущего слайда дает следующий результат:

Утверждение 1. *Текст.*

Лемма 2. *Второй текст.*

Утверждение 3. *Третий текст.*

Лемма 5. *Четвертый текст?*

Да, вложенные и общие счетчики можно совмещать, если того требует сложная структура вашего документа. При этом с самими счетчиками можно при этом работать с помощью методов, обсуждавшихся в прошлый раз.

Подключение файлов

Подключение tex-исходников

Когда кода становится слишком много, становится оправданно разделять его на отдельные файлы, и собирать документ в `main.tex` (который в идеале не должен содержать нетривиального кода). Ниже — пример типичного файла `main.tex`.

```
\input{preamble}

\begin{document}
    \input{titlepage}
    \input{chapter1}
    \input{chapter2}
    \input{chapter3}
\end{document}
```

input vs. include

Синтаксис подключения tex-файлов имеет вид `\input{filename(.tex)}`. При компиляции код из файла `filename.tex` подставляется вместо соответствующей команды. Его альтренатива — `\include{filename}`.

`input`	`include`
• Подставляет текст непосредственно	• Начинает текст с новой страницы и производит еще некоторые манипуляции
• Может быть вложенным	• Не может быть вложенным
• Чтобы не компилировать часть файлов — убирать или комментировать эти строки	• Чтобы не компилировать часть файлов — можно добавить `\includeonly{name1, name2}` в преамбуле

При использовании вложенного обращения к файлам (например, `\input` внутри файла, подключенного через `\input`) следует помнить, что иерархия файловой системы ведет отсчет из *корневой папки*, содержащей `main.tex`.

Подключение pdf-файлов

Подключение pdf-файлов производится с помощью пакета `pdfpages` и имеет, например, такой синтаксис:

```
\usepackage{pdfpages}

\begin{document}
    \includepdf[pages={1, 3, 5-6}]{filename.pdf}
\end{document
```

Пакет tikz-cd

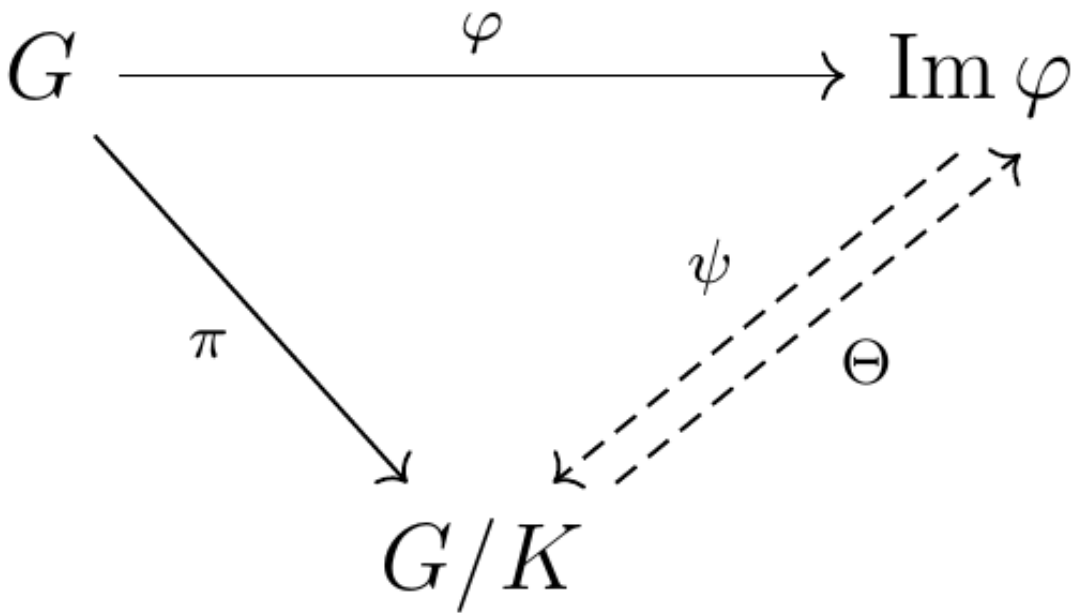
Пример использования tikz-cd

Набор пакетов `tikz` предоставляет инструменты для создания в texe разнообразной векторной графики. Для работы с коммутативными диаграммами есть специальный пакет `tikz-cd`. Ограничимся примером его использования.

```
\[
\begin{tikzcd}[row sep = huge]
    G \arrow{rr}{\phi} \arrow[swap]{dr}{\pi} & &
    \im\phi \arrow[dashrightarrow, swap]{dl}{\psi}
    \\
    & G / K \arrow[dashrightarrow, swap]{ur}{\Theta} &
\end{tikzcd}
\]
```

Пример использования tikz-cd

Код с предыдущего слайда дает следующий результат:



Графика с пакетом tikz I

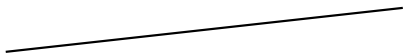
Введение

Нужно всего лишь...

```
\usepackage{tikz}
```

И теперь можно рисовать! Например, прямую линию...

```
\tikz \draw (0pt, 0pt) -- (1in, 8pt);
```



... или оранжевый кружок.

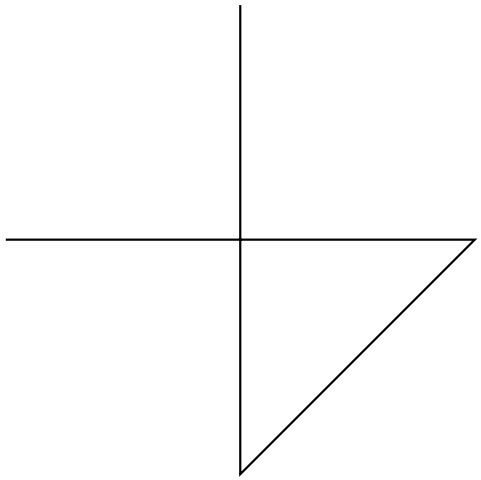
```
\tikz \fill[orange] (1ex, 1ex) circle (1ex);
```

У самурая нет цели, только путь

Путь — это основной блок всех рисунков в `tikz`. Он состоит из точек `(x,y)` и прямых `--`. Весь код помещается в окружение `tikzpicture`, а основной командой для рисования является команда `\draw`.

```
\begin{tikzpicture}
\draw (-1.5,0) -- (1.5,0) -- (0,-1.5) -- (0,1.5);
\end{tikzpicture}
```

Этот код дает следующий результат:



tikz vs. tikzpicture

Вообще говоря, для рисунков имеет место такая запись:

```
\begin{tikzpicture}
    % some code
\end{tikzpicture}
```

И такая запись:

```
\tikz % some code;
```

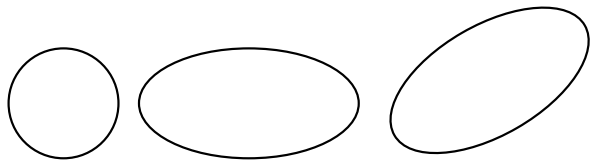
Они эквиваленты. Для «однострочных» второе может казаться привлекательнее, но на практике, конечно, таких коротких рисунков не будет. Поэтому первый способ предпочтительнее, но я для экономии места на слайде иногда буду писать `\tikz{...}`.

Кружочки

Нарисовать круг или эллипс — не просто, а очень просто.

```
\tikz \draw circle (10pt);
\tikz \draw ellipse (20pt and 10pt);
\tikz \draw [rotate=30] ellipse (20pt and 10pt);
```

Этот код дает следующий результат:



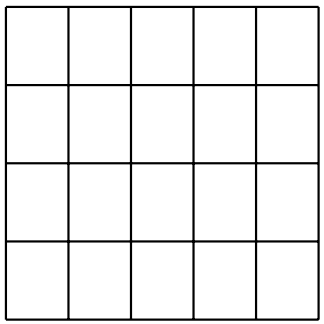
Как видим, второй эллипс повернут на 30°, что контролируется аргументом `rotate` со значением 30.

Сетка

Нарисовать сетку можно так:

```
\tikz \draw [xstep=0.4, ystep=0.5] (0,0) grid (2,2);
```

Этот код дает следующий результат:

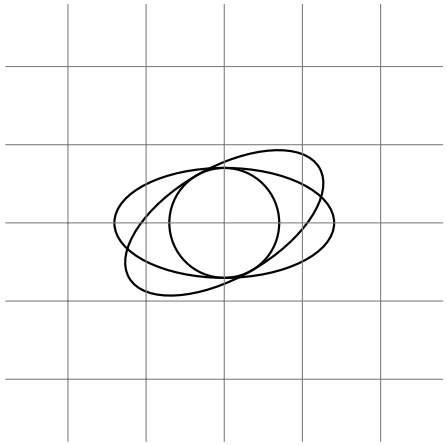


Есть, конечно, и просто аргумент `step`.
Кстати, перед тем как рисовать объект, можно указать точку «в которой» его нужно рисовать.

Комбинация объектов

```
\draw (0,0) circle (10pt);
\draw (0,0) ellipse (20pt and 10pt);
\draw [rotate=30] (0,0) ellipse (20pt and 10pt);
\draw [step=.5cm, gray, very thin] (-1.4, -1.4) grid (1.4,1.4);
```

Этот код дает следующий результат:

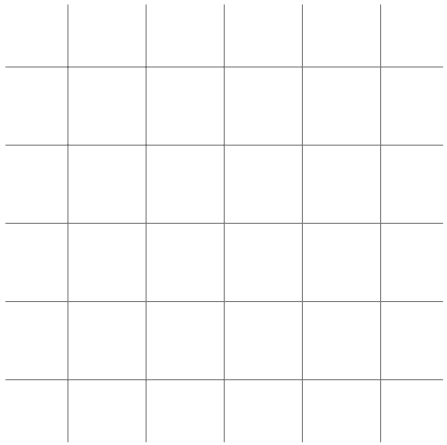


Упрощаем себе жизнь

На самом деле сетка, выполняет роль *вспомогательных линий*, и целых два аргумента (`gray, very thin`) говорят нам об этом. А ведь вообще вспомогательные линии — очень популярная штука. Можно сделать свой *стиль* для них:

```
\tikzset{help lines/.style={very thin, gray}}
\tikz [step=.5cm, help lines] (-1.4, -1.4) grid (1.4, 1.4);
```

Этот код дает следующий результат:

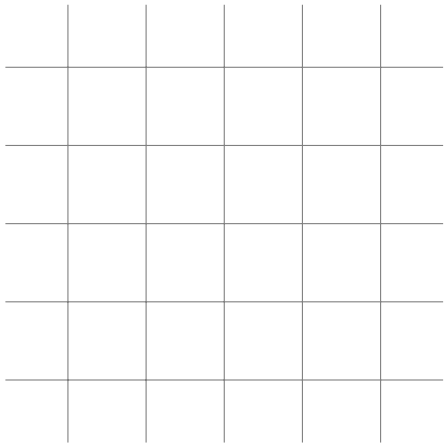


Величие — в стилях

Можно выстраивать иерархию стилей и передавать свои параметры, например, так:

```
\tikzset{help lines/.style={very thin, color=#1!50},
         help lines/.default={black}}
\tikzset{help grid/.style={step=#1, help lines=black},
         help grid/.default={0.5cm}}
      \tikz \draw [help grid] (-1.4, -1.4) grid (1.4,1.4);
```

Этот код дает следующий результат:



Параметров стиля может быть и несколько.

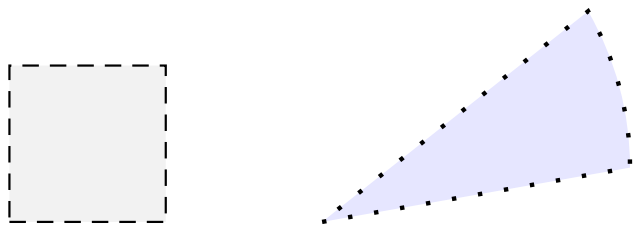
Опции рисунка

- Толщина:
  - ultra/very thin
  - thin
  - semithick
  - thick
  - ultra/very thick
- Цвет:
  - gray, reb, blue
  - `{rgb,255: red,21; green,66; blue,128}`
  - Микс: `red!10!blue` (10% красного и 90% синего)
  - Прозрачность: `green!50`
- Заполнение линии:
  - loosely/densely dashed
  - loosely/densely dotted

Управление опциями

```
\begin{tikzpicture}
\draw [fill = gray!10, thin, densely dashed]
(0,0) -- (1,0) -- (1,1) -- (0,1) -- cycle;
\draw [fill = blue!10, thick, loosely dotted]
(2,0) -- +(10:2) arc (0:30:2) -- cycle;
\end{tikzpicture}
```

Этот код дает следующий результат:



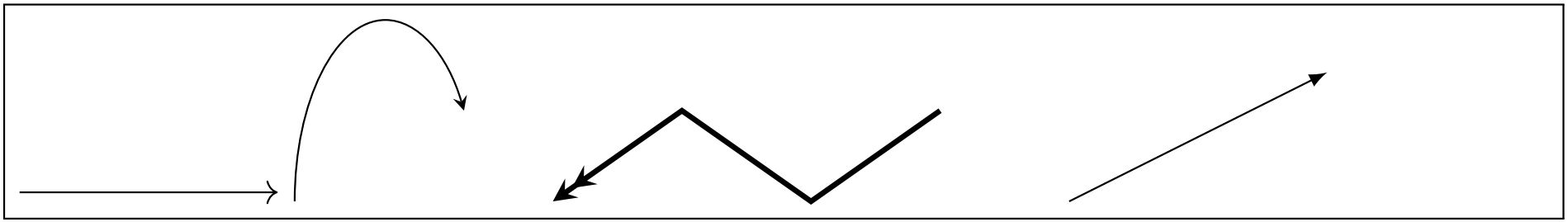
Можно задавать полярные координаты в виде `(angle: radius)`, можно в пути указывать координаты относительно последней точки, используя `+(...)`.

Стрелочки

Обычные стрелочки выглядят некрасиво, поэтому можно передать аргументом для всего окружения >=stealth.

```
\tikz [->] (0,0) -- (2, 0);
\begin{tikzpicture}[scale=2,>=stealth]
    \draw[->] (0,0) arc (180:30:10pt and 20pt);
    \draw[-latex] (3,0) -- +(1,1);
    \draw[<<-, very thick] (1,0) -- (1.5cm,10pt)
                                 -- (2cm,0pt)
                                 -- (2.5cm,10pt);
\end{tikzpicture}
```

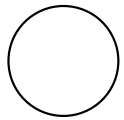
Этот код дает следующий результат:



Точки

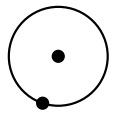
Можно задавать положение точки и переиспользовать его:

```
\tikz \coordinate (A) at (0,0);
\tikz \draw (A) -- (1,0);
```



Можно делать очень удобные вещи, упрощая себе вычисления:

```
\begin{tikzpicture}
    \coordinate (A) at (0,0);
    \coordinate (B) at (0.1,0.3);
    \node[draw,circle through=(A)] at (B) {};
\end{tikzpicture}
```



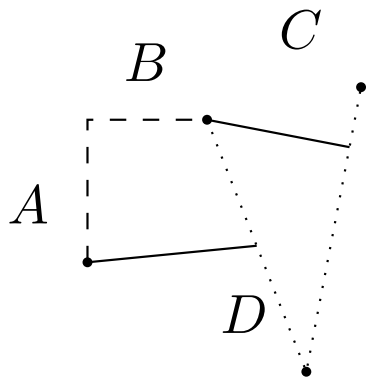
Для использования ключевого слова `through` в преамбуле требуется команда `usetikzlibrary{through}`.

Графика с пакетом tikz II

Упрощаем вычисления I

Вычислять положения многих объектов можно автоматически. Для этого в преамбуле потребуется команда `usetikzlibrary{calc}`. Дополним код с предыдущего слайда:

```
...
\coordinate (B) at (60: 1.5);
\coordinate (D) at ($(A) + (2, -1)$);
\draw [dashed] (A) -- (A |- B) -- (B);
\draw [dotted] (B) -- (D) -- (C);
\draw (B) -- ($(D)!(B)!(C)$); % высота
\draw (A) -- ($(B)!0.5!(D)$); % медиана
```



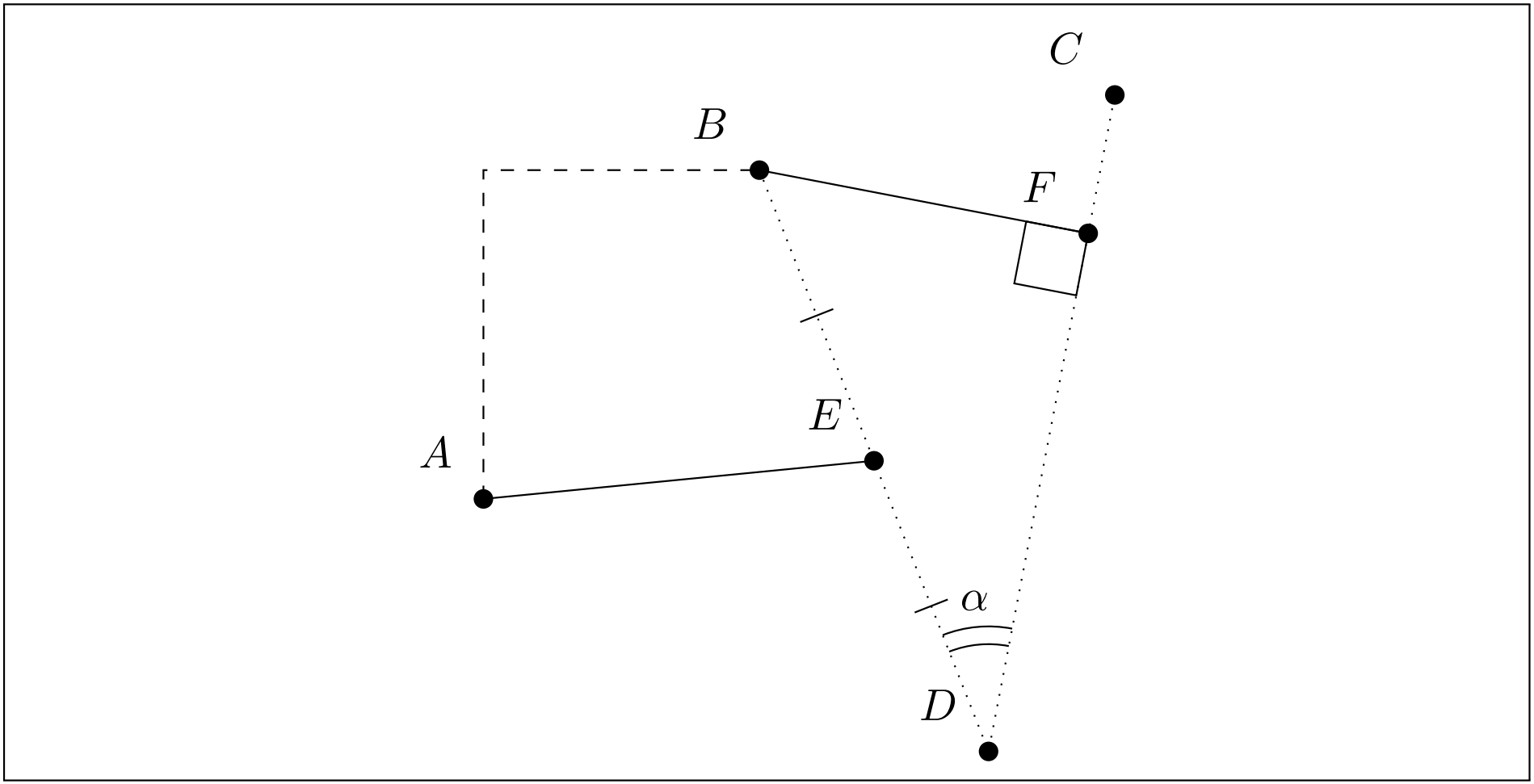
Углы и отрезки

Пакет `tkz-euclide` позволяет работать с отрезками, углами и другими объектами евклидовой геометрии. Снова дополним код с предыдущего слайда:

```
...
\tkzMarkAngle[mark=,arc=ll,size=10pt](C,D,B);
\tkzMarkRightAngle(D,F,B);
\tkzLabelAngle[pos=0.6](C,D,B) {$\alpha$};
\tkzMarkSegment[mark=|](E,B);
\tkzMarkSegment[mark=|](E,D);
```

Углы и отрезки

Окончательный код с предыдущих слайдов дает следующий результат:



Циклы

Оказывается, в texe циклы. Например, с их помощью можно провернуть такое:

```
\begin{equation*}
    r =
    \sqrt{\foreach \x in {a, ..., g} {
        \ifthenelse{\equal{\x}{a}}{}{+}
        \x^2
    }}
\end{equation*}
```

Этот код дает следующий результат:

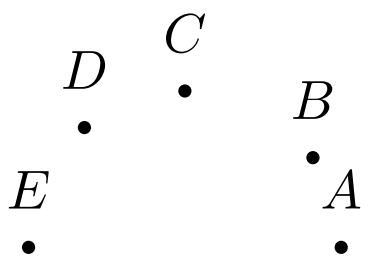
$$r = \sqrt{a^2 + b^2 + c^2 + d^2 + e^2 + f^2 + g^2}$$

Циклы в tikz

Удобство циклов становится очевидным в tikz.

```
\begin{tikzpicture}
    \def\r{1}
    \coordinate (A) at (0:\r);
    ... % объявление точек
    \coordinate (F) at (180:\r);
    \foreach \p in {A, ..., D} {
        \draw [fill=black] (\p) circle(1pt);
        \node [label=above:$\p$] at (\p) {};
    }
\end{tikzpicture}
```

Этот код дает следующий результат:

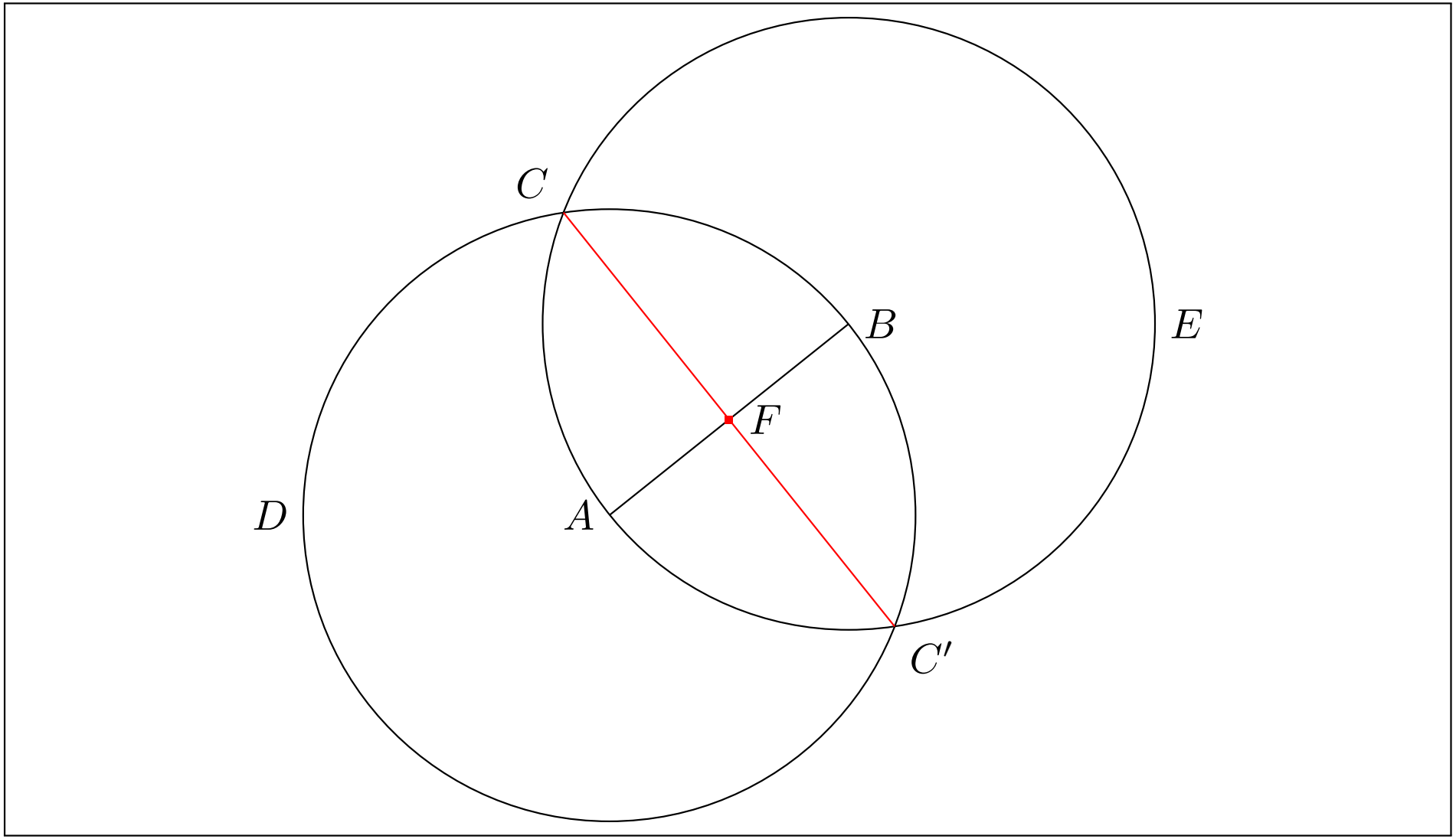


Упрощаем вычисления II

Рассмотрим более сложный пример автоматических вычислений и научимся именовать автоматически созданные объекты.

```
... % объявление точек
\draw [name path=A--B] (A) -- (B);
\node (D) [name path=D, draw, circle through=(B),
    label=left:$D$] at (A) {};
\node (E) [name path=E, draw, circle through=(A),
    label=right:$E$] at (B) {};
\path [name intersections={of=D and E,
    by={[label=above:$C$]C,
    label=below:$C'$]C'}}];
\draw [name path=C--C', color=red] (C) -- (C');
\path [name intersections={of=A--B and C--C', by=F}];
```

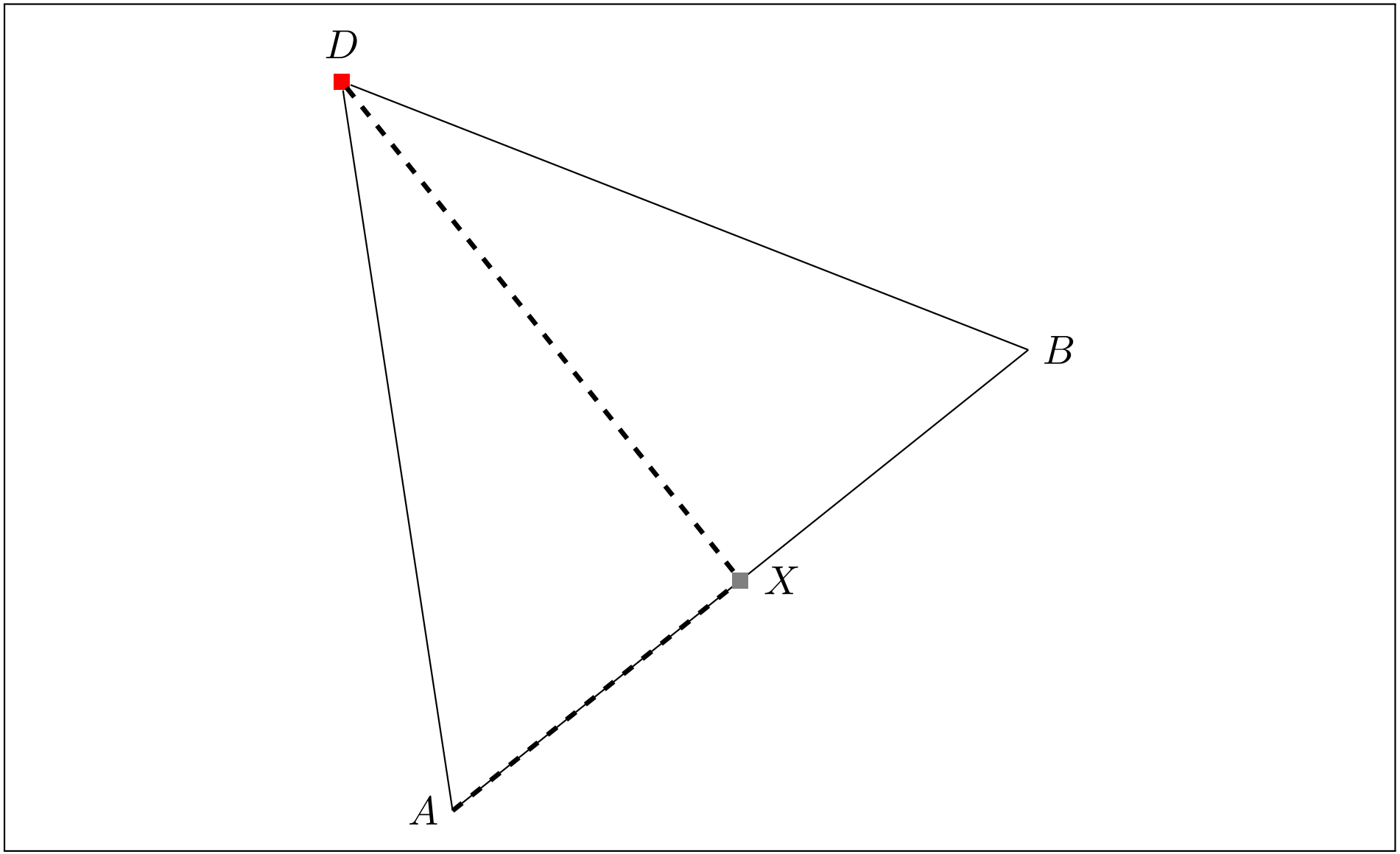
Код на предыдущем слайде дает следующий результат:



Упрощаем вычисления III

Рассмотрим еще один пример.

```
\begin{tikzpicture}
\coordinate [label=left:$A$] (A) at (0, 0);
\coordinate [label=right:$B$] (B) at (5, 4);
\draw (A) -- (B);
\node [fill=red, inner sep=2pt, label=above:$D$]
    (D) at ($ (A) ! .5 ! (B) ! {sin(60)*2} ! 90:(B) $) {};
\draw (A) -- (D) -- (B);
\node [fill=gray, inner sep=2pt, label=right:$X$]
    (X) at ($ (A) ! .5 ! (B)$) {};
\draw [very thick, dashed] (A) -- (X);
\draw [very thick, dashed]
    (X) -- ($(X) ! {sin(60)*2} ! 90:(B)$);
\end{tikzpicture}
```


D
B
X
A

Что дальше?

Великое множество пакетов

Есть масса полезных пакетов, о которых нет смысла рассказывать на курсе, потому что они решают свои конкретные задачи, которые могут у вас и не возникнуть (а еще потому, что их десятки и десятки). Но любой хороший пакет имеет понятную документацию! Упомянем некоторые из них:

- `multicol` — написание текста в несколько колонок
- `color` — гибкая настройка цветов
- `tocloft` — гибкая настройка страницы содержания
- `listings` — визуализация кода на языках программирования в tex-документах
- `algorithm2e` — описание алгоритмов
- `wrapfig` — размещение «плавающих» объектов (изображений, таблиц)

Целый интернет

LaTeX — очень старая технология, люди по всему миру пользуются ей десятилетиями. Если у вас никак не получается решить некоторую проблему, не бойтесь гуглить и ходить даже по старым форумам (хотя большинство ваших проблем наверняка решит StackOverflow).

Даже если найденное вами решение устаревшее, если вы понимаете, что вам нужно использовать его всего раз, — используёте, скорее всего это ничего не испортит.

Желаем удачи!

Всё!