Документация к пакету геометрических макросов macros.mp

Арсений Акопян
(akopjan@gmail.com)

Аннотация

Базируется на макросах украденых у М. Вялого, И. Богданова, А. Полозова и В. Родионова. Воровство в процессе.

Базовые команды

Расстояние между $z0$ и $z1$:
```
abs(z0-z1);
```

Повернуть точку $z1$ вокруг точки $z0$ на угол a:
```
z1 rotatedaround (z0, a);
```

Отразить точку $z0$ относительно прямой проходящей через точки $z0$ и $z1$:
```
z0 reflectedabout(z1, z2);
```

Обрезать пусть p с помощью отрезков $z0z1$ и $z1z2$:
```
p cutbefore (z0-z1) cutafter (z1-z2);
```

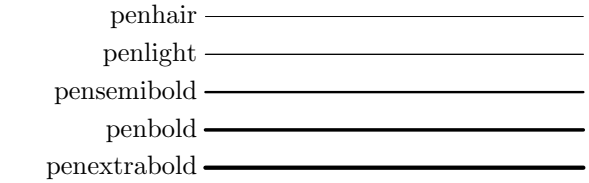
Длина пути p:
```
length p;
```

Дополнительный сдвиг буквы при пометке:
```
dotlabel.adjusted.top(btex X etex, z0 adjust (0, 0.5));
```

Настройки линий

Ширина

Указываем ширину линий `draw p penhair;`



Как правило **penlight** это ширина линии по умолчанию.

Пометки

Отметки символов

Ставим букву X, рядом с точкой $z0$:
```
label(btex $X$ etex, z0);
```

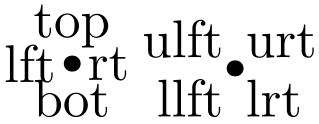
Ставим букву X, рядом с точкой $z0$, также отмечаем саму точку:
```
dotlabel(btex $X$ etex, z0);
```

Ставим букву X с белым фоном, рядом с точкой $z0$:
```
whitelabel(btex $X$ etex, z0);
```

Ставим букву X с белым фоном, рядом с точкой $z0$, также отмечаем саму точку:
```
whitedotlabel(btex $X$ etex, z0);
```

Расположить под отмечаемой точкой:
```
whitedotlabel.bot(btex $X$ etex, z0);
```

Другие расположения:



Отметки

За радиус дужки отвечает параметр `angle_radius`. За размеры засечек и расстояния между дужками `marksize`.

Квадратик, помечающий прямой угол $z1$:
`mark_rt_angle(z0, z1, z2);` Если фактически угол $z1$ не прямой, то строится ромбик.

Штрихованный квадратик, помечающий прямой угол $z0z1z2$:
```
markdashed_rt_angle(z0, z1, z2);
```

Квадратик с заданным размером, помечающий прямой угол $z0z1z2$:
```
mark_rt_angle_withsize(z0, z1, z2, 10);
markdashed_rt_angle_withsize(z0, z1, z2, 10);
```
Последний параметр это величина отметки. Если она меньше 0, то используется величина `angle_radius`.

Дуги помечающие угол $z0z1z2$. Угол должен быть меньше 180°. Последний параметр радиус, если он отрицательный, то подставляется величина `angle_radius`.

```
arcs(z0, z1, z2, 10);
arcs2(z0, z1, z2, 10);
arcs3(z0, z1, z2, 10);
```

Тоже самое, что и `arcs()`, но можно ставить пометку углу:
```
labelarcs(z0, z1, z2, 10, btex $\alpha$ etex);
```

Тоже самое, что и `labelarcs()`, но пометка окружается белым фоном:

```
whitelabelarcs(z0, z1, z2, 10, btex $\alpha$ etex);
```

Тоже самое, что и `labelarcs()` и `whitelabelarcs()` но указывается на каком расстоянии от вершины угла находится пометка:

```
labelarcsprof(z0, z1, z2, 10, 12, btex $\alpha$ etex);
whitelabelarcsprof(z0, z1, z2, 10, 12, btex $\alpha$ etex);
```

Прерывистые дуги помечающие угол $z0z1z2$. Последний параметр радиус, если он отрицательный, то подставляется величина `angle_radius`.

```
dashedarcs(z0, z1, z2, 10);
dashedarcs2(z0, z1, z2, 10);
dashedarcs3(z0, z1, z2, 10);
```

Тоже самое, что и `dashedarcs()`, но можно ставить пометку углу:

```
labeldashedarcs(z0, z1, z2, 10, btex $\alpha$ etex);
```

Тоже самое, что и `labeldashedarcs()`, но пометка окружается белым фоном:

```
whitelabeldashedarcs(z0, z1, z2, 10, btex $\alpha$ etex);
```

Тоже самое, что и `labeldashedarcs()` и `whitelabeldashedarcs()` но указывается на каком расстоянии от вершины угла находится пометка:

```
labeldashedarcsprof(z0, z1, z2, 10, 12, btex $\alpha$ etex);
whitelabeldashedarcsprof(z0, z1, z2, 10, 12, btex $\alpha$ etex);
```

Помеченные дуги для угла $z0z1z2$. Последний параметр радиус, если он отрицательный, то подставляется величина `angle_radius`.

```
mark_angle(z0, z1, z2, 10);
mark_angle2(z0, z1, z2, 10);
mark_angle3(z0, z1, z2, 10);
```

Залитая дужка угла $z0z1z2$. Последний параметр радиус, если он отрицательный, то подставляется величина `angle_radius`. За цвет отвечает `anglecolor`.

```
fill_angle(z0, z1, z2, 10);
```

Данная пометка помечает позволяет ставить на путях римские цифры от 0 до 5 и 10. Можно задать несколько путей, например `rimmark(p1, p2)`. Сам путь при этом не рисуется.

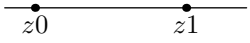
```
rimmark0(p);
rimmark(p);
rimmark2(p);
rimmark3(p);
rimmark4(p);
rimmark5(p);
rimmark10(p);
```


Данная пометка, тоже что и предыдущее, только окружает отметку белыми фоном:

```
whiterimmark0(p);
whiterimmark(p);
whiterimmark2(p);
whiterimmark3(p);
whiterimmark4(p);
whiterimmark5(p);
whiterimmark10(p);
```

Рисование линий

Рисовать сразу несколько линий:
```
Draw p1, p2, p3;
```

Линия с хвостиками за концевыми точками, сама линия при этом не рисуется, это только путь:
```
ddline(z0, z1)(0.2, 0.4);
```


Если линия уже задана как путь p. То чтобы не искать концы, можно написать так:
```
ddrealline(p)(0.2, 0.4);
```

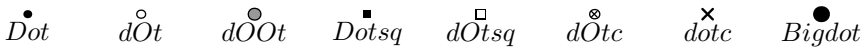
Дуга с концами в $z0$ и $z2$ и проходящая через $z1$:
```
arc(z0, z1, z2);
```

Дуга с хвостиками и концами в $z0$ и $z2$ и проходящая через $z1$:
```
ddarc(z0, z1, z2)(10, 20);
```
Параметры указываются в обычных градусах.

Часть окружности p находящаяся между точками $z0$ и $z1$, (против часовой стрелки):
```
cutcircle(p, z0, z1);
```

Отметка точек

Стандартный способ поставить точку в $z0$:
```
dotlabel(, z0);
```
Кроме того, можно использовать следующие отметки (это макросы), которые работают для нескольких точек:



Окружности и прямые

У всех окружностей внутренняя длина равна 4, в отличии от **fullcircle**, где она равна 8.

Окружность с центром в $z0$ и радиуса r:
```
circle(z0, r);
```
Радиус можно задавать и вектором:
```
circle(z0, z1-z2);
```

Окружность проходящая через токи $z0$, $z1$ и $z2$:
```
circumcircle(z0, z1, z2);
```

Пересечение двух прямых, первая из которых проходит через точки $z0$ и $z1$, а вторая $z2$ и $z3$:
```
crosspoint(z0, z1)(z2, z3);
```

Пересечение двух путей (прямых, окружностей и парабол) $p0$ и $p1$:
```
cross(p0, p1);
```
Можно использовать параметры `.first`, `.second`, `.top`, `.bot`, `.lft` и `.rt`.

Касательная из точки $z0$ к окружности $p0$:
```
support(p0, z0);
```

Возвращается точка касания. Можно использовать параметры `.first`, `.second`, `.top`, `.bot`, `.lft` и `.rt`. Работает также и с параболой.

Внешняя касательная к двум окружностями $p0$ и $p1$.

```
dbltangent1(p0, p1);
dbltangent2(p0, p1);
dbltangent11(p0, p1);
dbltangent12(p0, p1);
dbltangent21(p0, p1);
dbltangent22(p0, p1);
```

Первая цифра отвечает за номер касательной, вторая указывает на какой из окружностей лежит точка. Если число однозначно, то возвращается сама касательная (как путь соединяющий точки касания).
Если таких двух касательных не существует, то выдается сообщение.

Внутренняя касательная к двум окружностями $p0$ и $p1$.

```
dblintangent1(p0, p1);
dblintangent2(p0, p1);
dblintangent11(p0, p1);
dblintangent12(p0, p1);
dblintangent21(p0, p1);
dblintangent22(p0, p1);
```

Первая цифра отвечает за номер касательной, вторая указывает на какой из окружностей лежит точка. Если число однозначно, то возвращается сама касательная (как путь соединяющий точки касания).
Если таких двух касательных не существует, то выдается сообщение.

Если точка $z0$ лежит на окружности с центром в $z2$. То найти вторую точку пересечения окружности с лучом $z0z1$ можно так:

```
secondpoint(z0, z1, z2);
```

Построить центр окружности вписанной в угол $z0z1z2$ (против часовой стрелки) и проходящую через точку $z3$ (которая обязана лежать внутри угла):

```
angle_circle_in(z0, z1, z2, z3);
angle_circle_out(z0, z1, z2, z3);
```

Инверсия относительно окружности с центром в $z0$, радиусом r:

```
inversion(z0, r)(p);
```

p может быть, точкой, прямой или окружностью.

Гомотетия p с центром в $z0$ и коэффициентом k:

```
scaleabout(z0, k)(p);
```

Коники и разные кривые

Эллипс с полуосями a и b и центром в начале координат:

```
ellipse_canonical(a, b);
```

Эллипс с фокусами $z.f1$ и $z.f2$ и проходящий через точку $z0$:

```
ellipseFFP(z.f1, z.f2, z0);
```

Эллипс с фокусами $z.f1$ и $z.f2$ и касающийся прямой $z0z1$:
```
ellipseFFT(z.f1, z.f2, z0, z1);
```

Гипербола с фокусами $z.f1$ и $z.f2$ и проходящая через точку $z0$. Длина гиперболы равна 88. Регулируется с помощью переменных `hyp_start` и `hyp_final`.
```
hyperbolaFFP(z.f1, z.f2, z0);
```

Гипербола с фокусами $z.f1$ и $z.f2$ и касающаяся прямой $z0z1$. Длина гиперболы равна 88. Регулируется с помощью переменных `hyp_start` и `hyp_final`.
```
hyperbolaFFT(z.f1, z.f2, z0, z1);
```

Дуга гиперболы $xy = c^2$. Внутренняя длина выходящей гиперболы равна 88, параметривзована градусами радиус вектора (от 1 до 89):
```
hyperbolaxy(c);
```

Правая и левая дуга гиперболы с полуосями a и b и центром в начале координат. Длина гиперболы равна 88. Регулируется с помощью переменных `hyp_start` и `hyp_final`.
```
hyperbola_canonical_positive(a, b);
hyperbola_canonical_negative(a, b);
```

Парабола заданная уравнением $y = ax^2 + bx + c$:
```
parabola_canonical(a, b, c)(e, l);
```
(e, l) — координаты точки $(1, 1)$.

Парабола с фокусом в точке $z0$ и директрисой $z1, z2$:
```
parabolaFD(z0, z1, z2);
```

Коника проходящая через пять точек:
```
fivepointsconic(z0, z1, z2, z3, z4);
```
В случае гиперболы возвращает только одну из дугу. Чтобы выбрать другую надо написать `fivepointsconic2(...)`.

Коника касающаяся прямых $z0z1$, $z2z3$, $z4z5$, $z6z7$ и $z8z9$:
```
inscribed_in_pentagon_conic(z0, z1)(z2, z3)(z4, z5)(z6,z7)(z8, z9);
```
В случае гиперболы возвращает только одну из дуг. Чтобы выбрать другую надо написать `inscribed_in_pentagon_conic2(...)`.

Точка с параметром k на пути p. Считается что путь имеет нормальную s и его длина равна 1.
```
pointonpath(p, k);
```

Элементы треугольника

Основание биссектрисы выходящей из вершины $z1$ треугольника $z0z1z2$:
```
bisector(z0, z1, z2);
```

Основание внешней биссектрисы выходящей из вершины $z1$ треугольника $z0z1z2$:
```
exbisector(z0, z1, z2);
```
Если основание получается слишком далеко, то выдается основание обычной биссектрисы повернутое на 90° вокруг $z1$.

Центр вписанной окружности треугольника $z0z1z2$:

```
incenter(z0, z1, z2);
```

Центр вневписанной окружности треугольника $z0z1z2$ соответствующий вершине $z0$:
```
excenter(z0, z1, z2);
```

Вписанная окружность треугольника $z0z1z2$:
```
incircle(z0,z1,z2);
```

Вневписанная окружность треугольника $z0z1z2$ соответствующая вершине $z0$:
```
excircle(z0,z1,z2);
```

Основание высоты треугольника $z0z1z2$ выходящей из вершины $z1$:
```
altitude(z0, z1, z2);
```

Ортоцентр треугольника $z0z1z2$:
```
orthocenter(z0, z1, z2);
```

Медиана треугольника $z0z1z2$ выходящая из вершины $z1$:
```
median(z0, z1, z2);
```

Центр тяжести треугольника $z0z1z2$:
```
centroid(z0, z1, z2);
```

Центр описанной окружности треугольника $z0z1z2$:
```
circumcenter(z0, z1, z2);
```

Окружность Эйлера треугольника $z0z1z2$:
```
euler_circle(z0, z1, z2);
```

Изогонально сопряженная точка точки $z3$ в треугольнике $z0z1z2$:
```
isogonal_point(z0, z1, z2)(z3);
```

Точка Нагеля треугольника $z0z1z2$:
```
nagel_point(z0, z1, z2);
```

Точка Жергона треугольника $z0z1z2$:
```
gergonne_point(z0, z1, z2);
```

Точка Лемуана треугольника $z0z1z2$:
```
lemoine_point(z0, z1, z2);
```

Точки Торичелли треугольника $z0z1z2$:
```
torricelli_point(z0, z1, z2);
torricelli_point2(z0, z1, z2);
```
По умолчанию, возвращается точка лежащая «внутри» треугольника.

Точки Апполония треугольника $z0z1z2$:
```
apollonius_point(z0, z1, z2);
apollonius_point2(z0, z1, z2);
```
По умолчанию, возвращается точка лежащая «внутри» треугольника.

Точки Брокара треугольника $z0z1z2$:

```
brocard_point(z0, z1, z2);
brocard_point2(z0, z1, z2);
```

Точка Фейербаха треугольника $z0z1z2$:
```
feuerbach_point(z0, z1, z2);
```

Педальная окружность точки $z3$ относительно треугольника $z0z1z2$:
```
pedal_circle(z0, z1, z2)(z3);
```

Чевианная окружность точки $z3$ относительно треугольника $z0z1z2$:
```
cevian_circle(z0, z1, z2)(z3);
```

Вписанная в треугольник $z0z1z2$ коника с перспектором в $z3$:
```
inscribed_in_triangle_conic(z0, z1, z2)(z3);
```

Разные кривые

Лемниската Бернулли с фокусами $z0$ и $z1$:
```
lemniscate_of_bernoulli(z0, z1);
```
Длина кривой равна 180. Параметризована полярным углом в узловой точке.

Кардиоида с каспом в $z0$ и вершиной в $z1$:
```
cardioid(z0, z1);
```
Длина кривой равна 360. Параметризована полярным углом в каспе.

Циссоида Диоклеса с каспом в $z0$ и основанием высоты опущенной на асимптоту в $z1$:
```
dissoid_of_diocles(z0, z1);
```
Длина кривой равна 120. Параметризована полярным углом в каспе.