Asymptote: Macros and Packages

Asymptote (Vector Graphics Language)

Getting Started - Basics - Drawing - Reference - Examples - Macros and Packages - Advanced Asymptote - 3D Graphics - Help

Useful functions - CSE5 Package - How to

Definitions

You can define your own functions in Asymptote. As an example, let's say you wanted to make a function called `newfunction` that takes a pair (a, b) and a real value r as input, and returns the pair $(a+r, b+r)$. In addition, you want it to simply return the pair (a, b) if no value of r is specified, so you want r to default to 0. The code would be as follows:

```
pair newfunction(pair z, real r=0)
{
 real a,b;
 a=z.x;
 b=z.y;
 return (a+r,b+r);
}
```

Put this definition in an asymptote document and then test it using some command like

```
draw(newfunction((20,30))--newfunction((20,30),30)--(0,0)--cycle);
```

See if it works!

Notice that the function must be declared a pair since it returns a pair, and each of the variables must be declared some data type too. The default value of r was set to 0 by $r=0$, and the actual function procedure goes in between `{}`. To define a function with no output simply put `void` before the function name. This is the general format for a function definition.

Packages

Asymptote comes with several packages that contain useful functions for various purposes. For example, the package `graph.asy` contains the function

```
Circle(pair p, real r, int n=400);
```

which is a more accurate circle (having 400 nodes by default) than the built-in `circle` command. To use this function and others in graph.asy, simply put the command

```
import graph;
```

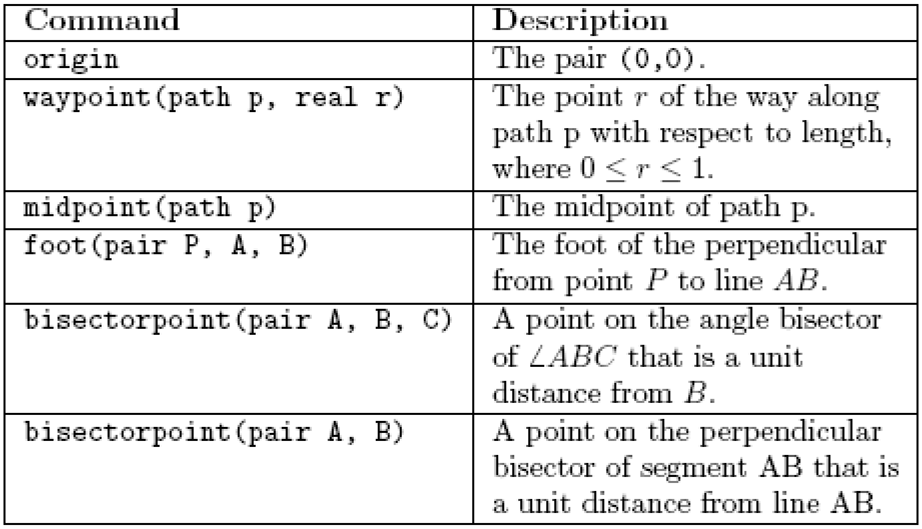
at the top of your Asymptote document. Graph also has more advanced functions such as the ability to Graph a function (http://artofproblemsolving.com/wiki/index.php?title=Asymptote:_Graphing)

You can create your own package by simply creating a new .asy file (say `MyMacros.asy`) with your own definitions in it, and saving it in the directory in which Asymptote is installed (`C:\Program Files\Asymptote` by default). Then `import MyMacros;` in your document, and you'll be set!

The Olympiad Package

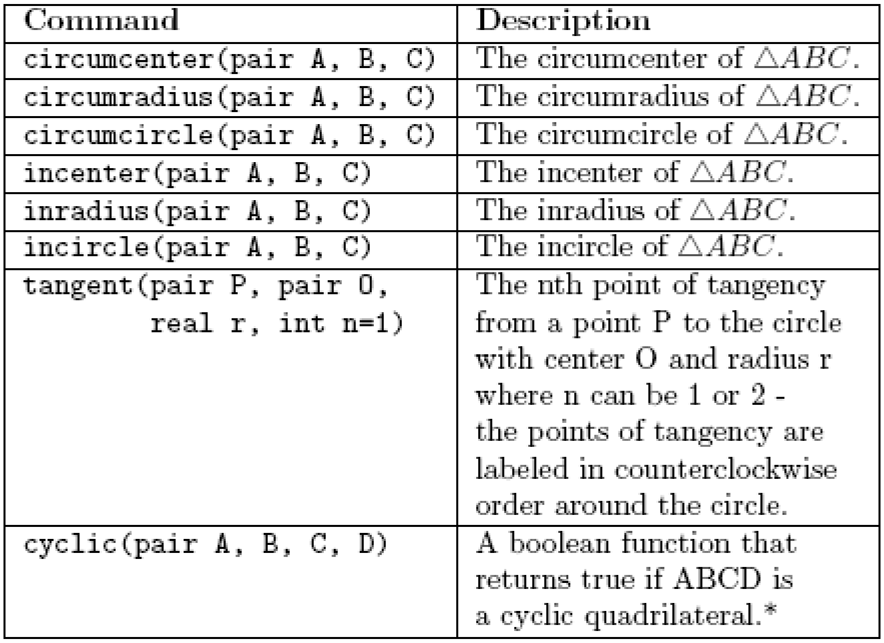
We have created an Olympiad package for Asymptote which includes macros for all the constructions that come up most often in Olympiad geometry problems! You can obtain the package olympiad.asy by clicking here (http://math.berkeley.edu/~monks/images/olympiad.asy) or here (http://www.artofproblemsolving.com/Forum/viewtopic.php?f=519&t=165767) (the latter link has a few usage examples).

This package includes the following definitions:



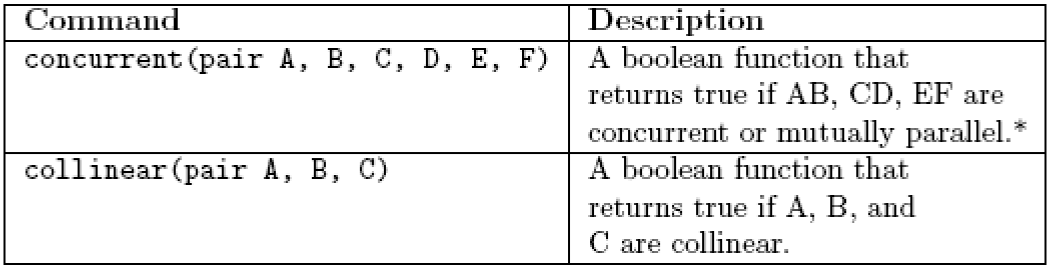
Command	Description
`origin`	The pair `(0,0)`.
`waypoint(path p, real r)`	The point r of the way along path p with respect to length, where $0 \leq r \leq 1$.
`midpoint(path p)`	The midpoint of path p.
`foot(pair P, A, B)`	The foot of the perpendicular from point P to line AB.
`bisectorpoint(pair A, B, C)`	A point on the angle bisector of $\angle ABC$ that is a unit distance from B.
`bisectorpoint(pair A, B)`	A point on the perpendicular bisector of segment AB that is a unit distance from line AB.

Table 1: Useful points defined in the olympiad.asy package



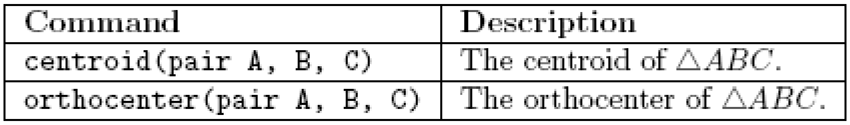
Command	Description
`circumcenter(pair A, B, C)`	The circumcenter of $\triangle ABC$.
`circumradius(pair A, B, C)`	The circumradius of $\triangle ABC$.
`circumcircle(pair A, B, C)`	The circumcircle of $\triangle ABC$.
`incenter(pair A, B, C)`	The incenter of $\triangle ABC$.
`inradius(pair A, B, C)`	The inradius of $\triangle ABC$.
`incircle(pair A, B, C)`	The incircle of $\triangle ABC$.
`tangent(pair P, pair O, real r, int n=1)`	The nth point of tangency from a point P to the circle with center O and radius r where n can be 1 or 2 - the points of tangency are labeled in counterclockwise order around the circle.
`cyclic(pair A, B, C, D)`	A boolean function that returns true if ABCD is a cyclic quadrilateral.*

Table 2: Circle-related definitions in the olympiad.asy package



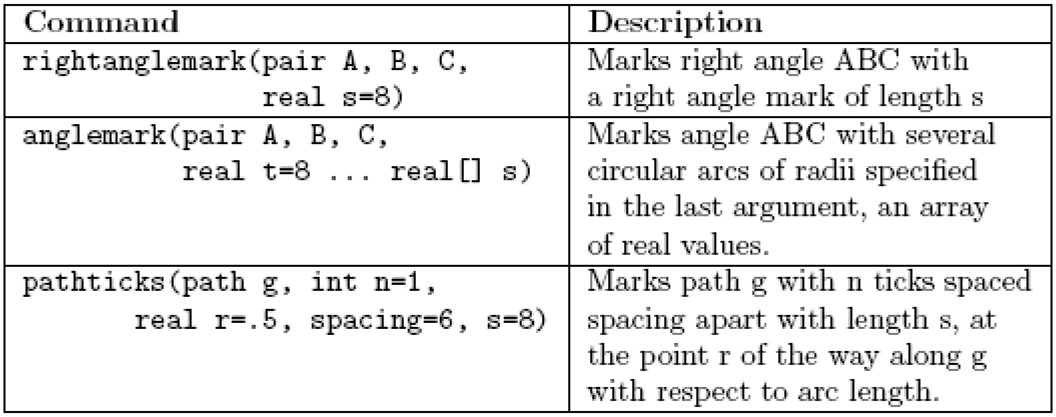
Command	Description
`concurrent(pair A, B, C, D, E, F)`	A boolean function that returns true if AB, CD, EF are concurrent or mutually parallel.*
`collinear(pair A, B, C)`	A boolean function that returns true if A, B, and C are collinear.

Table 3: Collinearity and concurrency from the olympiad.asy package



Command	Description
`centroid(pair A, B, C)`	The centroid of $\triangle ABC$.
`orthocenter(pair A, B, C)`	The orthocenter of $\triangle ABC$.

Table 4: Triangle-related definitions in the olympiad.asy package



Command	Description

`rightanglemark(pair A, B, C,`
`real s=8)`
	Marks right angle ABC with a right angle mark of length s

`anglemark(pair A, B, C,`
`real t=8 ... real[] s)`
	Marks angle ABC with several circular arcs of radii specified in the last argument, an array of real values.

`pathticks(path g, int n=1,`
`real r=.5, spacing=6, s=8)`
	Marks path g with n ticks spaced spacing apart with length s, at the point r of the way along g with respect to arc length.

Table 5: Tick marks and angle marks in the olympiad.asy package

Note: A sequence of variables without type declarations indicates that they are the same type as the variable preceding it. For example, the notation `concurrent(pair A, B, C, D, E, F)` indicates that all of the variables should have type pair.

* These boolean functions test for equality within 10^{-5} ps points in order to avoid approximation errors.

Next: Advanced

Retrieved from "https://artofproblemsolving.com/wiki/index.php?title=Asymptote:_Macros_and_Packages&oldid=105480"