

Maple: Plotting Points, Lines and Circles

© Philip B. Yasskin, Texas A&M Univ., 1997-2006

1. A point in the plane is specified by giving two numbers (x,y) . The first number x gives the position right or left of the origin (right if x is positive and left if x is negative). The second number y gives the position above or below of the origin (above if y is positive and below if y is negative). In Maple a point is denoted by two numbers in square brackets $[x,y]$. Thus the point $(2,5)$ is typed as $[2,5]$. To enter a list of points, the points must be separated by commas and enclosed in another set of square brackets. For example, type the following list of four points and press **Enter**. (Every command must end with a semi-colon (`;`) which shows the output or a colon (`:`) which hides the output.)

```
> [ [1,2], [2,1], [3,1], [4,2] ];
```

If you typed it correctly, Maple will repeat what you typed. If you made a mistake, Maple will put the cursor where it thinks you made the mistake. Just fix it and press **Enter** again. If you forgot the semi-colon, Maple will give you a warning. Just backspace, add the semi-colon and press **Enter** again. **DO NOT RETYPE the whole line.**

2. The Maple `plot` command will plot a list of points and connect them with straight lines like doing a dot-to-dot picture. For example, if you draw a line from $(1,2)$ to $(2,1)$ to $(3,1)$ to $(4,2)$, you will get a U shape. To see this, click on the previous line and modify it to look like the following and press **Enter**. **DO NOT RETYPE.**

```
> plot([ [1,2], [2,1], [3,1], [4,2] ], color=red);
```

The `plot` command needs parentheses (`)` around the list of points and any options such as a choice of `color`.

3. You can save a plot for future use by storing it in a memory location. Click at the beginning of the previous line and add `you:=` and click at the end of the line and change the semi-colon to a colon. **DO NOT RETYPE the whole line.** The line should now look like:

```
> you := plot([ [1,2], [2,1], [3,1], [4,2] ], color=red):
```

The `you:=` stores the plot in a memory location named “you” for future use. When you press **Enter** you will not see the plot because the colon prevents a bunch of junk from being printed out. To see the plot again, type

```
> you;
```

4. Try this again. Plot an octagon:

```
> octagon := plot([ [1,0], [4,0], [5,1], [5,4], [4,5], [1,5],  
[0,4], [0,1], [1,0] ], color=blue):  
> octagon;
```

5. If you have several lists of points, enclose them in square brackets `[]`. Try the following:

```
> line1 := [ [1,3], [2,3] ];  
> line2 := [ [3,3], [4,3] ];  
> lines := plot([line1, line2], color=[green, magenta]):  
> lines;
```

Colors can be specified for each piece.

6. Once you have several plots, you can put them together using the `display` command: (That's a "one" after the "p".)

```
> with(plots):  
> p1 := display([you, octagon, lines]):  
> p1;
```

What did you get?

7. At this point you should save your file so you don't lose it. To do this, click on **File** and **Save** and save it as `face`. Remember to save it frequently.

8. Now try:

```
> diamond := [ [1,3], [1.5,3.5], [2,3], [1.5,2.5], [1,3] ];  
> diamondline := plot([diamond, line2], color=[green, magenta]):  
> p2 := display([you, octagon, diamondline]):  
> p2;
```

9. You can now display your two pictures as a movie:

```
> display([p1, p2], insequence=true);
```

Click in the plot. Then in the control bar, click on the **Loop** and **Play** buttons. You can slow it down or speed it up in the control bar. The option `insequence=true` makes the pictures show sequentially instead of at the same time.

10. Next add a circle of radius `.25` centered at `[2.5,2]`:

```
> with(plottools):  
> nose:=circle([2.5,2], .25, color=orange):  
> p3:=display({p1, nose}):  
> p4:=display({p2, nose}):  
> display([p1, p2, p4, p3], insequence=true, scaling=constrained,  
axes=none);
```

The option `scaling=constrained` makes horizontal and vertical distances equal. The option `axes=none` eliminates the axes. The order that the frames are listed (`p1, p2, p4, p3`) controls the order they are shown.

11. You can also rotate and translate your plots. For instance, the following command rotates the plot `p2` clockwise by $\frac{\pi}{4}$ radians about the point `[2.5,2.5]` (the center of the face) and then translates it 2 units to the right and 1 unit down.

```
> p5 := translate( rotate(p2, -Pi/4, [2.5,2.5]), 2, -1):
```

Rotating and translating `p4, p3` and `p1` also, we get another movie:

```
> p6 := translate( rotate(p4, -Pi/2, [2.5,2.5]), 4, -2):  
> p7 := translate( rotate(p3, -3*Pi/4, [2.5,2.5]), 6, -1):  
> p8 := translate( rotate(p1, -Pi, [2.5,2.5]), 8, 0):  
> display([p1, p5, p6, p7, p8, p7, p6, p5], insequence=true,  
scaling=constrained, axes=none);
```

12. Finally, you can save your movie as an animated `gif` file, so that you can include it on a web page. To do this, right click in the plot and select **Export As >> GIF**. Save it as `face.gif`. View your movie by finding it in Windows Explorer and double clicking on it.

13. You are now on your own. Try making the other eye blink. Change the `circle` command to `disk`. Add hair or ears or teeth. Or try to make the mouth talk. Or design your own movie.