Now we get to some of the good stuff. This lesson deals with 3 graphics statements. They are about lines, starting points, ending points and pensizes. The rest of the concepts in the program should be familiar to you.

If you don't have a compiler .. just read the code and think about what the lines mean. The comments should help. If you do havea compiler ...try it out. You can either cut and paste this into your editor or print it out and then type it in line by line. I recommend that you read it over good once..then cut and paste...nothing wrong with that ... cutting and pasting is a feature that takes some of the drudgery out of programming on the mac. Beware though that cutting and pasting bits and pieces here and there can cause you more time and trouble than its worth.. but then what the hay... you will be forced to figure it out and thats good too...Is there nothing about this machine that doesn't have a silver lining?

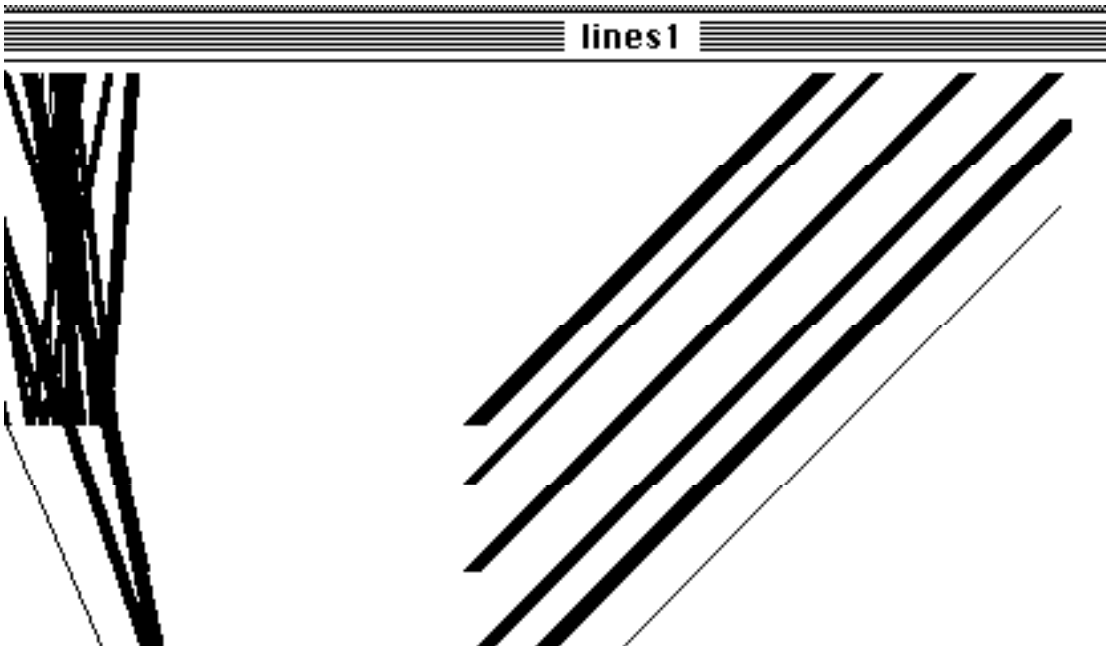
The new control statements are **moveto, lineto**, and **pensize.** Thats easy enough, huh? Here it is:

```
program lines1;                                  {program name}

uses memtypes,quickdraw,fixmath;                 {units used}

var
x,y,z,ps,start,endpt:integer;                    {variables identifed/defined}

begin                                            {begin program}

for x:= 1 to 65 do                               {condition, do this until x =65}

begin                                            (begin loop for x}
x:= x+ 1;                                        {initialize x with a value of x+ 1}
y:= 300;                                         {initialize y with a value of 300,    this is the
location on the screen of y}
ps:=random mod 6;                                {pensize will be random}
start:= random mod 30;                           {startpoint will be random}
endpt:= random mod 100;                          {endpoint will be random}
pensize(ps,ps);                                  {pensize two dimensions}
moveto (start,start);                            {move the pen to a location}
lineto (endpt,endpt);                            {draw a line to a second location}
moveto (x,y);                                    {move the pen to a location}
lineto (endpt,endpt);                            {draw a line to a second location}
moveto (x,y);                                    {move the pen to a location}
lineto (endpt,endpt);                            {draw a line to a second location}
```

```
moveto (400,50);                {move to location 400,50}
lineto (175,275);               {draw a line to 175,275}
readln;                         {take a look before pressing <CR>}
end;                            {end of loop}
end                             {end of program}
```

This program doesnt do much. It has no practical purpose except to illustrate the use of these statements. Notice that we have **USED** two significant **Units**: **quickdraw** and **fixmath**. **Quickdraw** lets us do graphics. **Fixmath** lets us do mathematical things fast (there is another math **UNIT** called **SANE**, it allows for very very precise math). **Units** are procedures and routines in the toolbox (rom) that let us program some things rather easily. We could probably do the same things without the **USE** of these **UNITS** but it would take a long time. In essense, we are calling on rom and invoking pre-written code to assist us. This is a reason that Mac programs have such similar interfaces and usability! Other programmers have been using the same **UNITS**. This will become more evident later.

Here is a snapshot of what the program does. It looks a bit different each time.



Notice the different size of the lines...thts due to the random size of the pen! Your not impressed, huh? Stay tuned..... it gets better.
