See how easy kindergarten can be. After this lesson you will know about such abstract things as **variables** and **constants** and why there were all those punctuation marks in the first lesson.

Let's take a look at the format of a pascal program. I goes something like this:

```
-----------------------------------------------------------------
PROGRAM Guess_the_number;                          {name of program)

USES Memtypes, QuickDraw, OSIntf, ToolIntf,PackIntf;

VAR
number, guess   : integer;

BEGIN

number:= random mod 5;                             {generates a random number    from 1-5}

writeln ('I am thinking of a number.");
writeln (The number is between 1 and 5.');
writeln ('What is it?');
readln(guess);                                     {reads the users entry}
If guess = number then
writeln ('FANTASTIC, YOU GUESSED THE NUMBER!");    {pat on the back for the            user}
else if guess < number then                        {if its lower than the       number}
writeln ('So sorry. You were too low. It was', number);
else
writeln ('So sorry. You were too high. It was', number);
readln;
end.
-----------------------------------------------------------------
```

This program may not win you friends or influence people, but you can impress someone in you neighborhood with it at least once. What it does is, as the comments say, it generates a random number from 1 to 5. It ask for a guess by the user, then compares it and responds accordingly.

Let's take a look at the program format. First the program name line.

```
PROGRAM Guess_the_number;                          {name of program)
```

Pascal program names do not allows for spaces. Use underlines when necessary (___). I havent found any other restrictions yet, but then i am only in the 3rd grade.

The curly brackets { } those have special significance they let you comment on any line without worrying about whther the compiler will mess you up by trying to understand what is inside of them. A word about comments, use them, often. By doing so you can glance quickly at a program and refresh your memory about what you were doing. With only 6-10 lines of code this may not seem important but as our programs get bigger it will.

Next comes a line tht begins with **USES.**

```
USES Memtypes, QuickDraw, OSIntf, ToolIntf,PackIntf;
```

'USES' will be explained in future lessons, for now just keep in mind that these are from the **TOOLBOX**, see there is the famous **QUICKDRAW!**

Notice that at the end of many lines there is a semicolon. The semicolon is the character that identifies the end of a line. (I have found circumstances where there is no semicolon and don't understand yet, why it isn't always required).

The next line is:

```
VAR
```

See, this line doesn't have a semicolon.

The following line after VAR idientifies the VARiables tht will be used in this program and what kind of variable they are.

```
number, guess   : integer;
```

Both number and guess are the variables that are used. they are both integers. (If you are like me you might not remember that math term. An INTEGER is a whole number without a fraction. 35 is a INTEGER, so is 52 but 6.14 is not an INTEGER. Get the idea. Don't worry I read up on it. By the way 6.14 is a REAL number.)

That could also have been written

VAR

```
number   :integer;                                    {note the colon separation}
guess    :integer;
```

Some values within a program never change. These values cannot be variables then, huh? RIGHT. These values aren't variable at all. They are **CONSTANTS.** We didn't use any constants in our program, but just so you won't forget whata constant is , here is an example:

```
CONST                                          {this would have   preceded the
VAR
                                               section. Constants         are
identified first}

days_in_year = 365;                            {the number of days       in the year
is now                                         set. It will not change      through
out the                                        program}
```

Next line is

```
BEGIN              {note that the begin statement doesn't have a semi     colon}
```

The next line is

```
number:= random mod 5;  {as commented this generates a random number from       1 to 5. Note
the colon and                                  equal sign after number.    This
INITIALIZES the                                variable.}
```

The rest of the program is self explanatory, I think. Let me know if it isn't and i will modify this for others that may pass this way. Well thats it. There is only one more lesson in the kindergarten class. Then a few easy questions for you to answer to yourself and you graduate. See it's not so hard, is it?

Whew! I am glad I didn't go into teaching . This is hard work. Good luck and work this one over too. Play with it. The worse that can happen is a system meltdown.