

```

% This program illustrates the use of the Matlab
% random number generator.

% First we test the simplest use by assigning a random number to x.
% Notice that the outputs are uniformly distributed between 0 and 1.
for n = 1:3
    x = rand
end;

control = input('Hit enter (or return) to continue');

% It is an interesting fact that we often want our 'random' numbers
% to be repeatable. For instance, we may rewrite code to be clearer,
% or more efficient, but the output is supposed to be identical to the
% original. This would be difficult to test if random data were not
% repeatable. Here is how Matlab handles this situation.

j = 1357;
rand('state',j);
for n = 1:3
    x = rand
end;

control = input('Hit enter (or return) to continue');

j = 1357;
rand('state',j);
for n = 1:3
    x = rand
end;

control = input('Hit enter (or return) to continue');

% Notice that the output from these two loops is identical.
% The value j is often called the 'seed' of the random number
% generator.

% The function rand can easily be used to generate random integers.
% Here are some random integers between 0 and 100. The outputs here
% are not quite equally likely; 0 and 100 are less likely than other integer
% outputs. Can you see why?
for n = 1:3
    x = int16(100*rand)
end;

% One often wants random numbers with a nonuniform distribution.

```

```
% Matlab provides Gaussian outputs with the function randn. Other
% distributions can be generated by transforming the variables.
% What do you think the distribution of y is if  $y = -\log(\text{rand})$  ?
```