One more Program/Tutorial on Random Numbers

How can a computer, which works with deterministic algorithms, where every step is fully predictable, produce ‘Random Numbers’? The underlying concept is explained in the program below:

```
/* This program generates N random numbers between 0 and 9 */
#include <stdio.h>
#include <time.h>
#include <stdlib.h>

int main(){

        int i,N, seed, nextno;
        printf("Enter number of random numbers you want ");
        scanf("%d",&N);
        printf("Enter 4 digit seed ");
        scanf("%d",&seed);

        for(i=0;i<N;i++)
        {
                if(seed >= 1000)
                  {seed=seed*8257/10000;}
                else
                  {seed=seed*8257/1000;}
                nextno=seed%10;
                printf(" next random number %d \n",nextno);
        }//for

return 0;
}//main
```

Comments:

[1] This is a very simple program to generate random single digit integers starting from some seed. It simply takes as seed a 4 digit number. It multiplies it by a fixed 4 digit number, which I just picked to be 8257. One can play around with it. It then throws away the last 4 digits or the last 3 digits thus always keeping the seed around 3 to 4 digits. The random number is the last digit of the seed.

[2] The point of the program is simply that if you multiply two large numbers together, the middle digit is 'random' for most purposes.

[3] When numbers are stored as real numbers with double precision, the decimal arithmetic has automatic trunction. If you look at the last significant digit stored, it would be essentially 'Random'. Trunction errors for most computations appear random.

[4] Which brings us to the question: What is a Random Number? May be we should ask what do we want in a sequence of Random Numbers? If the sequence of numbers satisfy some desired properties, they may be good enough for our purpose. Here are some properties, we might want in our random numbers:

1. No number should be favored over others. If we generate enough random digits, all digits 0 through 9 should arise roughly equal number of times. Note that this is a necessary but not a sufficient criterion for randomness. Why? Consider the sequence 0,1,2,3,.... which repeats after every 10 numbers.

2. We must also require that every time we get 0 the next number should be equally likely to be 0 through 9. Similarly, if the first number is 1 the next number should be equally likely to be 0 through 9. And so on. What this criterion means is that if we pair consecutive pairs of numbers in a sequence (i,j) — the pairs must be equally distributed among 100 possibilities.

3. Does this criterion guarantee we have random numbers? The answer is No. Once again, we can simply generate these 100 pairs in order and repeat. That would not be Random. One must have a similar criterion for every 3-tuple. Every 4-tuple, etc. To be truly random, the sequence would have to satisfy an infinite number of such criterion.

4. However, in practice, one can easily take a crude random number sequence and shuffle it a few times (using the crude random number generator) and that would make a very good

random number generator for all purposes.

5. Shuffling is a factorial process and factorial grows very rapidly with number. Shuffling among 'sort of random' numbers makes them 'highly random'. Say you have a sequence that is kind of random but repeats after a million numbers. If you take 100 of those at a time and randomly pick one out of them as your next number. Then add the new number in the sequence to your list and continue. You would end up with a sequence that would never repeat on any computer, and will satisfy most criterion for randomness.