

Randomness may be effective, since a random element is chosen from a table that is randomly generated. On the other hand, memory consumption may be quite huge.

The random generator [1...256] I have found on White Flame's page:

```

        lda seed
        beq doEor
        clc
        asl
        beq noEor      ;if the input was $80, skip the EOR
        bcc noEor
doEor   eor #$1d
noEor   sta seed

```

Now, in order to get random number in a range e.g. between 1 and 6 you can build a table of 256 bytes that is accessed by index with the previously generated random number

```

        tax
        lda rnd,x

rnd     .byte $4,$1,$2,$2,$5,$5,$6,$1
        .byte $3,$1,$2,$4,$1,$2,$4,$4
        ...
        .byte $3,$4,$4,$6,$2,$5,$2,$3

```

Such a table of random numbers in TAsm-Format may be generated with a python script:

```

from random import Random

range_lo = 1
range_hi = 7 # ..6
bytes = 256
byte_per_line = 8

g = Random(42) # initialize Wichmann Hill seed
r = ''
cnt = 0
for i in range(1,bytes,1):
    cnt = cnt + 1
    r = r + '$' + hex(g.randrange(range_lo,range_hi)).lstrip('0x') + ','
    if ((cnt % byte_per_line) == 0):
        print '        .byte ' + r.rstrip(',')
        r = ''

```

However, this method has a major drawback: You can't map a set of 256 numbers to an arbitrary range of numbers while keeping the same probability for each number. For example, when you want to get values from 0 to 156 then it is obvious that some numbers would appear twice in your table, while others appear just once. This has of course a huge impact on the randomness of your results. A better, although much slower way to get random numbers in a chosen interval is:

1. extend the above described mechanism to create 24-bit random numbers
2. multiply the random number with your max. number
3. use the highbyte of the result

From:  
<https://codebase64.org/> - Codebase 64 wiki

Permanent link:  
[https://codebase64.org/doku.php?id=base:fast\\_8bit\\_ranged\\_random\\_numbers&rev=1429237909](https://codebase64.org/doku.php?id=base:fast_8bit_ranged_random_numbers&rev=1429237909)

Last update: **2015-04-17 04:31**

