

# Re: Binary Shuffling

---

*Source:* <http://sci.tech-archive.net/Archive/sci.physics/2007-11/msg01745.html>

---

- *From:* Douglas Eagleson <[eaglesondouglas@xxxxxxxx](mailto:eaglesondouglas@xxxxxxxx)>
  - *Date:* Mon, 26 Nov 2007 15:10:26 -0800 (PST)
- 

On Nov 25, 7:15 pm, The Ghost In The Machine  
<[ew...@xxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:ew...@xxxxxxxxxxxxxxxxxxxxxxxxxxxx)> wrote:

In sci.physics, DouglasEagleson  
<[eaglesondoug...@xxxxxxxx](mailto:eaglesondoug...@xxxxxxxx)>  
wrote  
on Sun, 25 Nov 2007 12:19:12 -0800 (PST)  
<[77b78e25-1f77-47e5-987d-2569daf5b...@xxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:77b78e25-1f77-47e5-987d-2569daf5b...@xxxxxxxxxxxxxxxxxxxxxxxxxxxx)>:

1 1 1 1 0 0 0 0

A four one word is given. It has also four zero's, making a binary word. A shuffle will make all the permutations of all the combinations. This particular combination being four 1's and four 0's.

My question how to shuffle the set?

How is a special shuffle constructed to shuffle all binary's!!!!

A special reference is needed, my thinking I guess. Because each duplicate 1 or 0 is a combination word, not a permutation word.

ABCDOIUY to replace 11110000 makes a purmutation

## Re: Binary Shuffling

One method is to use a pick-and-delete, a destructive method that picks items and deletes them from the input, while setting them in the output.

This can be implemented as follows (in C):

```
char inp[8] = {'1','1','1','1','0','0','0','0'};
```

```
int genshuffle( char out[8])
{
for(int i = 8; i > 1; i--)
{
int r = makerand(i);
out[i-1] = inp[r];
for(int j = r+1; j < i; j++)
inp[j-1] = inp[j];
}
out[0] = inp[0];
}
```

where makerand(n) generates a random integer from 0 to n-1; the implementation thereof is left to the interested reader.

Various modifications are easily done, among them first copying the input to the output, then picking the desired bit in the output and swapping.

If you're more interested in enumerating all possible shuffles into a list, one can count from 0 to 11110000 and select all numbers containing