Consider the following $O(n)$ algorithms to shuffle an array:

```java
public static void shuffle1(int[] A){
    Random randomGenerator = new Random();
    for (int i=0;i<A.length;i++){
        int r = randomGenerator.nextInt(A.length);
        int t = A[i];
        A[i]=A[r];
        A[r]=t;
    }
}

public static void shuffle2(int[] A){
    Random randomGenerator = new Random();
    for (int i=0;i<A.length;i++){
        int r = i+randomGenerator.nextInt(A.length-i);
        int t = A[i];
        A[i]=A[r];
        A[r]=t;
    }
}
```

1. Will either or both produce all permutations of $A$ with equal probability?

2. For an array of length $n$, how many different combinations of random numbers will shuffle1 generate? Will each of them result in a different permutation of the original array?

3. For an array of length $n$, how many different combinations of random numbers will shuffle2 generate? Will each of them result in a different permutation of the original array?

4. For each method, compute the probability of obtaining each of the six permutations of the array $A = \{1, 2, 3\}$.