The goal of this note is to design and analyse an efficient randomized algorithm for the following problem.

**Input:** an array $A$ of the length $2n$ such that exactly $n$ elements are $X$ and exactly $n$ elements are $Y$.

**Output:** The number $i$ (at least one) such that $A[i] = X$.

There is a straightforward deterministic algorithm solving this problem

The function $\text{FindXDeterministic}(A[1..2n])$ is defined as:

```plaintext
function FindXDeterministic(\&;A[1..2n])
    for $i$ from 1 to $n + 1$
        if $A[i] == X$
            return $i$
    end for
end function
```

Its worst-case complexity (i.e., the number of operations in the worst case) is $\Theta(n)$.

Consider the following randomize algorithm

### Algorithm 1 Randomized algorithm

The function $\text{FindX}(A[1..2n])$ is defined as:

```plaintext
function FindX(\&;A[1..2n])
    for $j$ from 1 to $n$
        $i \leftarrow \text{random}(1, 2n)$
        if $A[i] == X$
            return $i$
    end for
    return $\text{FindXDeterministic}(A)$
end function
```

This algorithm first tried to find a desired index “by chance” and if it does not succeed, it calls a deterministic algorithm. In the worst case, the number of operations will be again $\Theta(n)$.

**Statement.** For every array $A$ of length $2n$ with $n$ $X$’s and $n$ $Y$’s, the expectation (i.e. the average) of the number of operations is $\Theta(1)$.

**Proof.** There exist constants $c_1$ and $c_2$ such that the number of operations performed by the call of $\text{FindXDeterministic}(A)$ is at most $c_1 n$ and the number of operations performed by every iteration of the first loop in Algorithm 1.

For every $1 \leq k \leq n$, we introduce event $B_k$ = “a desired index was found on $k$-th iteration of the first loop”. We also introduce event $C$ = “a desired index was not found by the first loop”.

- The probability of $B_1$ is equal to $\frac{1}{2}$, because exactly half of indices from 1 to $2n$ would give us a result.
- If the result was not found at the first iteration (this happens in the half of cases), it will be found in the half of cases at the next iteration, so $\Pr(B_2) = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$.
- There are only $\frac{1}{4}$ of the cases, when the answer is not found by neither of the first two iterations. In the half of these cases, a result will be found by the third iteration, so $\Pr(B_3) = \frac{1}{4} \cdot \frac{1}{2} = \frac{1}{8}$.
- In general, the probability of the fact that a result will not be found by the first $k - 1$ iterations is $\frac{1}{2^{k-1}}$. In the half of the cases, the $k$-th iteration will find a result, so $\Pr(B_k) = \frac{1}{2^{k-1}} \cdot \frac{1}{2} = \frac{1}{2^k}$.
- Analogously, $\Pr(C) = \frac{1}{2^n}$
If event $B_k$ happens, then the number of operations is $c_2 k$, if $C$ happens, then the number of operations is at most $c_1 n + c_2 n$. Then the expectation of the number of operations is equal to

$$c_2 \Pr(B_1) + c_2 \Pr(B_2) + \ldots + nc_2 \Pr(B_n) + (c_1 n + c_2 n) \Pr(C) = c_2 \left( \frac{1}{2} + \frac{2}{2^2} + \frac{3}{2^3} + \ldots + \frac{n}{2^n} \right) + \frac{c_1 n + c_2 n}{2^n}.$$

The first summand can be found as follows

$$\frac{1}{2} + \frac{2}{2^2} + \frac{3}{2^3} + \ldots + \frac{n}{2^n} = \frac{1}{2} + \left( \frac{1}{2^2} + \frac{1}{2^2} \right) + \left( \frac{1}{2^3} + \frac{1}{2^3} + \frac{1}{2^3} \right) + \ldots + \left( \frac{1}{2^n} + \ldots + \frac{1}{2^n} \right) =$$

$$\left( \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \ldots + \frac{1}{2^n} \right) + \left( \frac{1}{4} + \frac{1}{8} + \ldots + \frac{1}{2^n} \right) + \ldots + \left( \frac{1}{2^n} + \frac{1}{2^n} \right) =$$

$$\left( 1 - \frac{1}{2^n} \right) + \left( \frac{1}{2} - \frac{1}{2^n} \right) + \ldots + \left( \frac{1}{2^n - 1} - \frac{1}{2^n} \right) = \left( 1 + \frac{1}{2} + \frac{1}{4} + \ldots + \frac{1}{2^n - 1} \right) - \frac{n}{2^n} =$$

$$2 - \frac{n + 2}{2^n}.$$

Hence the expectation can be written as

$$c_1 \left( 2 - \frac{n + 2}{2^n} \right) + \frac{(c_1 + c_2) n}{2^n} = \Theta(1).$$

□