```
import numpy as np
#generate one random number
print(np.random.rand())
#generate an array with 1 random number
print(np.random.rand(1))
#generate an array with 10 random numbers
print(np.random.rand(10))
#generate an array with 3 rows and *2 columns
print(np.random.rand(3,2))
```

Similar to how LCG generates a fixed stream of random numbers given a fixed x_0 [assuming a, m, c are fixed]. Mersenne Twister will give a fixed stream of random numbers given a starting point of Random State. This RandomState will evolve every time a random number is generated (just like how x evolves). A RandomState contains the following information.

- the string ‘MT19937’, specifying the Mersenne Twister algorithm.
- a 1-D array of 624 unsigned 32-bit integers `keys`.
- an integer `pos`.
- an integer `has_gauss`.
- a float `cached_gaussian`.

In the previous cell, the first time `np.random.rand()` is called, a Random State will be created and used to generate the first random number. Throughout the cell, this Random State then updated to the next RandomState in a deterministic way whenever a random number is generated.

We can use `get_state()` to check what is stored in a state.

```
import numpy as np

np.random.rand()

print(np.random.get_state())
```