Robbing a safe

We are modelling an attempt to open a bank safe (for nefarious purposes).

The **key** part is to open the safe. We have a set of stolen keys, one of them guaranteed to open the safe–but we do not know which one.

There are N keys (consider N = 100) and it takes 10 seconds to try a key. The safe is protected by an automatic alarm that will sound after 10 wrong keys are tried.

If the alarm is triggered, security guards will arrive in 3 minutes; we must avoid meeting them face–to–face, so we must start running 20 seconds before they arrive.

Task: write the code of that handles the event reporting a **key mismatch**.

```
FAILURE:
    n++ ;     // this will be the n-th attempt (int n = 0 ;)
    end = Now + 10 ;    // in seconds
    if( Success( n ) )
        EQ.Insert( end , SUCCESS ) ;
    else
        EQ.Insert( end , FAILURE ) ;

bool Success( int n )
{
    return drand48() < 1.0 / (N − n ) ;
}
```

How does it work?

```
EQ.Insert( 0 , FAILURE ) ;
while( (E = EQ.Deletemin()) != NULL ) {
    Now = E->Time ;
    switch( E->Type ) {
        case FAILURE:
        case SUCCESS:
        case ROBBED:
        case RUN:
    }
}
```

The subtleties must be incorporated:

```
FAILURE:
    n++ ;               // this will be the n-th attempt
    if( n == 11 ) {
        Thriller( Now ) ;
    else {
        end = Now + 10 ;    // in seconds
        if( Success( n ) )
            EQ.Insert( end , SUCCESS ) ;
        else
            EQ.Insert( end , FAILURE ) ;
    }
    break ;
```

Thriller

```
SWEAT:
   if( Now + 10 + Robtime > Arrival )
      EQ.Insert( Now , RUN ) ;
   else {
      n++ ;
      end = Now + 10 ;
      if( Success( n ) )
         EQ.Insert( end , SUCCESS ) ;
      else
         EQ.Insert( end , SWEAT ) ;
   }
   break ;
```

Some thinking helps

The probability that the first key opens the safe is $\frac{1}{\mathsf{N}}$. The probability that the $\mathsf{k}-th$ key opens the safe is:

$$P(\mathcal{X}=\mathsf{k}) = \frac{\mathsf{N}-1}{\mathsf{N}} \times \frac{\mathsf{N}-2}{\mathsf{N}-1} \times ... \times \frac{\mathsf{N}-\mathsf{k}}{\mathsf{N}-(\mathsf{k}-1)} \times \frac{1}{\mathsf{N}-\mathsf{k}}$$

Number on the lucky key

$$P(\mathcal{X} = \mathsf{k}) = \frac{1}{\mathsf{n}}$$

So, if I want to select a **variate** representing the lucky key, what I need is a uniformly distributed integer between **1** and **n** which can readily be obtained by:

```
1 + (int)(N * drand48())
```

```
key = 1 + (int)(N * drand48()) ;
if( key < 11 )
    Breeze() ;
else if( key < 26 ) // or is it 25?
    Sweat() ;
else
    RunEmptyHanded() ;
```