2 process software synchronization

- Lamport (Bakery algorithm)
Bakery Algorithm (1974)

```plaintext
choosing[i] = 1;
num[i] = max(num[0], num[1], ..., num[n-1]) + 1;
choosing[i] = 0;
for j = 0 to n-1 {
    while (choosing[j] == 1) <no-op>;
    while (num[j] <> 0 and (num[j], j) < (num[i], i)) <no-op>;
}
Critical Section
num[i] = 0;
```

waiting for
num to clear

space to change
Choose $i = 4 - 1$

while $i > 0$

for $j = i, i + 1, \ldots, n - 1$

Check if $F(x_j, y_j) = 2$

Choose $x_j = 1$
proof 2 case for loop

\[ \text{num}[i] = \text{temp} + 1 \]

Combine numer (all num) => temp

\[ \text{num}[i] = \text{max (all num) + 1} \]
Eekel

CS

Please call INT.

method I → for unprocedural only

head were shallow.
1. Study in kennel code
2. Decrease in pain (e.g., taxol has decreased)
3. Multilobe cinn do not work
   (for V. short.cs.)

Discussion
- Exchange INTL Xchg
- Compare & Study
- FastSet <- Academic
- (b) Draw Step
  (c) Test Read & write in one
  Need diverse intersharing
- Multi-CPU system
Set \( x \) to 1

value was
what else did
you give us

\[ \text{return (true)} \]

\[ \begin{align*}
\text{return } & \text{(true)} \\
\text{if } & \text{true } x \\
\text{else } & \text{false } x
\end{align*} \]

\[ \text{if } x \neq \text{false } x \]

\[ 1 \leq x \leq r \]
\[ I = \int_{-\infty}^{\infty} f(x) \, dx \]

\[ p \xrightarrow{\alpha=0} \]

\[ q \xrightarrow{\alpha=0} \]
\[ x \text{ s.t. } x \in \{ 0, 1 \} \text{ s.t. } \exists y \in \{ 0, 1 \} \text{ s.t. } t(x, y) \]
int test_and_set (int new_value, int *lock_pointer);

.globl test_and_set

test_and_set:

# label
movl 4(%esp),%eax
# get new_value into %eax
movl 8(%esp),%edx
# get lock_pointer into %edx
xchgl %eax,(%edx)
# swap %eax with what is stored in (%edx)
# and don't let any other cpu touch that
# memory location while you're swapping
ret
# return the old value that's in %eax
What is the meaning of round-up? 

- Test case 
- Discussion