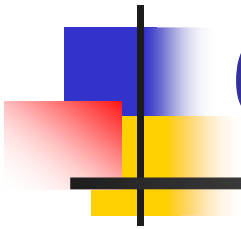
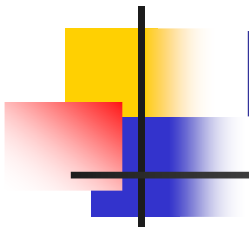


6.033 Spring 2007

Lecture 5

Clients and Servers within a Computer

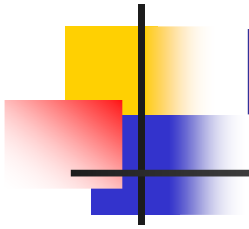


Bounded Buffer Code

```
Proc send(p, m) {
   while true do {
        if (p.in-p.out < N) then {
           p.buffer[p.in mod N] ß  m;
           p.in ß  p.in+1;
           return;}
   }}
```

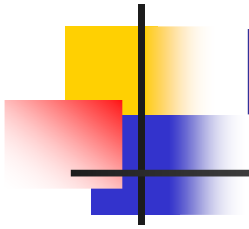
Bounded Buffer Code

```
Proc receive(p) returns msg {
  while true do {
      if (p.out < p.in) then {
        var m ß  p.buffer[p.out mod N];
         p.out ß  p.out+1;
        return(m);}
}}
```



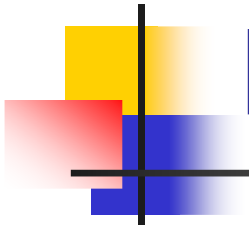
Bounded Buffer Code

```
Proc send(p, m) {
   while true do {
        acquire(p.lock);
       if (p.in-p.out < N) then {
          p.buffer[p.in mod N] ß   m;
          p.in ß  p.in+1;
          return;}
   }}
```



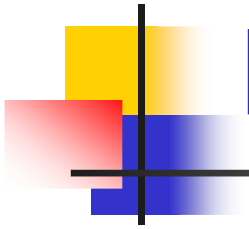
Bounded Buffer Code

```
Proc send(p, m) {
   while true do {
         acquire(p.lock);
        if (p.in-p.out < N) then {
           p.buffer[p.in mod N] ß  m;
           p.in ß  p.in+1;
            release(p.lock);
           return;}
   }}
```



Bounded Buffer Code

```
Proc send(p, m) {
   while true do {
        acquire(p.lock);
       if (p.in-p.out < N) then {
          p.buffer[p.in mod N] ß  m;
          p.in ß  p.in+1;
           release(p.lock);
          return;}
        release(p.lock);}}
```



Bounded Buffer Code

```
Proc receive(p) returns msg {
   while true do {acquire(p.lock);
        if (p.out < p.in) then {
           var m ß  p.buffer[p.out mod N];
           p.out++;
            release(p.lock) ;
           return(m);}
         release(p.lock)}}
```