

Unicus feras Arruda.

4,0

① Para isso, utilizarei o mecanismo send/receive, onde send é não bloqueante e receive é bloqueante.

Assinatura das funções send e receive:

```
send(Who* to, char* message); // to e from são  
receive(Who* from, char* message); // referências para  
// os clientes.  
// se NULL, o campo  
// é ignorado
```

```
void down()  
{  
    send(Server, "Down");  
    receive(NULL, NULL);  
}
```

Server é a referência para o único servidor existente. O esquema de Port é utilizado.

```
void Up()  
{  
    send(Server, "Up");  
}
```

Em down(), é utilizado NULL pois o único processo que vai me enviar um send é o Server. A mensagem não é necessária visto que já é sabido que está me dando o vez.

// Código do servidor

void simulaMutex ()

{

int mutex = 1;

Who c, tmp;

char * msg;

Queue Q;

while (1)

{

receive (&c, msg);

if (msg == "Down")

{

if (mutex == 1)

{

mutex = 0;

send (c, "Your turn");

}

else

{

Q.insert(c);

}

}

else if (msg == "Up")

{

if (Q.isEmpty())

{

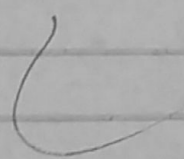
mutex = 1;

}

else

{

0, 1



```
tmp = Q.getFirst();  
send(tmp, "Your turn");
```

```
}
```

```
}
```

```
}
```

```
}
```

✓