Ping-Pong-Spieler in C-Linda: zwei abwechselnd aktive Prozesse

```
ping(...)
{ while(...)
  { out("ping");  /* spiele ping */
    in("pong");   /* warte auf pong */
  }
}

pong(...)
{ while(...)
  { in("ping");
    out("pong");
  }
}
```

Parallele Berechnungen auf Vektoren

```
/* ein Prozess fuer jede Komponente */
for (i=1; i<=n; i++) eval(f(i));
...
f(int i)
{ in("V",i,?v);
  out("V",i,compute(v));
}
```

Dinierende Philosophen

Philosoph:

```
phil(int i)
{ while(true)
  { think();
    in("stab", i);
    in("stab", (i + 1) % 5);
    eat();
    out("stab", i);
    out("stab" (i + 1) % 5);
  }
}
```

Initialisierung:

```
for(i = 0; i < 5; i++)
{ out("stab", i);
  eval(phil(i));
}
```

Client-Server-Kommunikation in Linda

Idee: numeriere Anfragen

```
server()
{ int i=1;
  out("serverindex", i);
  while(true)
  { in("request", i, ?req);
    ...
    out("response", i, result);
    i++;
  }
}

client()
{ int i;
  in("serverindex", ?i); /* hole Auftragsnummer */
  out("serverindex", i + 1);
  ...
  out("request", i, req); /* in req steht der eigentliche Auftrag */
  in("response", i, ?result);
  ...
}
```
