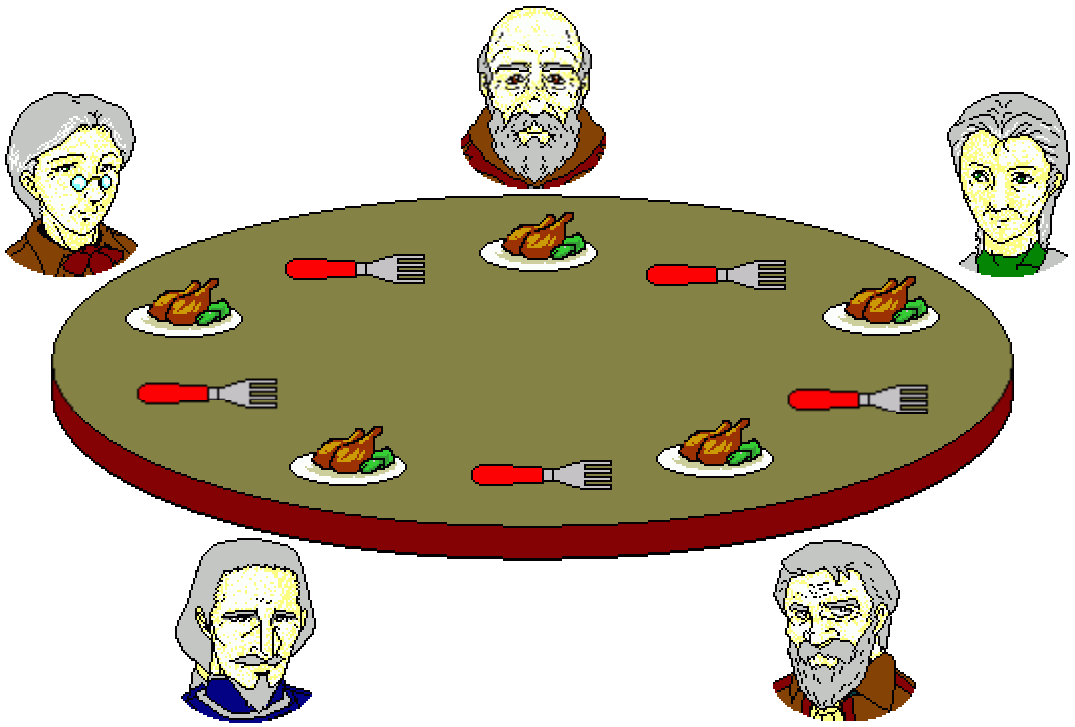
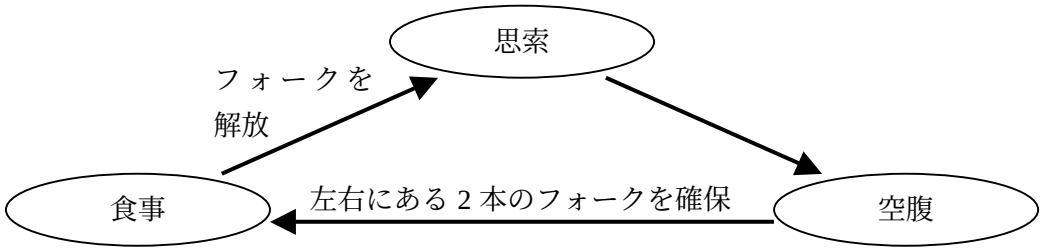
食事をする哲学者問題 (Dining Philosophers Problem)



- 5 人の哲学者がテーブルについており、5 本のフォークが哲学者の間に 1 本ずつ置かれている。
- 哲学者は 思索 → 空腹 → 食事 → 思索 を繰り返す。
- 空腹となった哲学者が食事をするためには、左右にあるフォークを 2 本とも確保する必要がある。
- フォークを確保できないときは待つ。ただし空腹のまま長時間待たされると餓死してしまう。



各哲学者の状態遷移

問題 ： 上記の条件で全ての哲学者を餓死させないようにするにはどうすればいいか？

解法1：繋がったフォーク

広域変数と初期値

```
semaphore fork := 1;
```

プロセス *i* (0≦*i*<5) **の処理**

```
while (true) {
   wait (fork);     // 全てのフォークを確保
   食事;
   signal (fork);   // 全てのフォークを解放
   思索;
}
```

解法 2: 左利きの哲学者

広域変数と初期値

```
semaphore fork [5] := {1, 1, 1, 1, 1};
```

プロセス 0 の処理

```
while (true) {
   wait (fork [1]); // 左フォーク確保
   wait (fork [0]); // 右フォーク確保
   食事;
   signal (fork [0]); // 右フォーク解放
   signal (fork [1]); // 左フォーク解放
   思索;
}
```

プロセス *i* (1≦*i*<5) **の処理**

```
while (true) {
   wait (fork [i]);              // 右フォーク確保
   wait (fork [(i+1) mod 5]);    // 左フォーク確保
   食事;
   signal (fork [(i+1) mod 5]); // 左フォーク解放
   signal (fork [i]);            // 右フォーク解放
   思索;
}
```

解法 3: 我慢する哲学者

広域変数と初期値

```
semaphore fork [5] := {1, 1, 1, 1, 1};
```

プロセス *i* (0≦*i*<5) **の処理**

```
while (true) {
   wait (fork [i]);                                  // 右フォーク確保
   while (try&wait (fork [(i+1) mod 5]) = false) { // 左フォーク確保できるまで繰り返す
      signal (fork [i]);                             // 右フォーク一旦解放
      少し待つ;
      wait (fork [i]);                               // 右フォーク再確保
   }
   食事;
   signal (fork [(i+1) mod 5]);                      // 左フォーク解放
   signal (fork [i]);                                // 右フォーク解放
   思索;
}
```