Version 1

```
program versionone;
int threadnumber;

threadone()
{
  while true do {
    while threadnumber == 2 do {} /* busy wait */
    criticalsectionone;
    threadnumber = 2;
    otherstuffone;
  }
}

threadtwo()
{
  while true do {
    while threadnumber == 1 do {} /* busy wait */
    criticalsectiontwo;
    threadnumber = 1;
    otherstufftwo;
  }
}

main()
{
  threadnumber = 1;
  pthread_create(tid, threadone);
  pthread_create(tid, threadtwo);
}
```

Version 2

```
program versiontwo;
int Thred1inside, Thred2inside;

threadone()
{
  while true do {
    while Thred2inside do {} /* busy wait */
    Thred1inside = 1;
    criticalsectionone;
    Thred1inside = 0;
    otherstuffone;
  }
}

threadtwo()
{
  while true do {
    while Thred1inside do {} /* busy wait */
    Thred2inside = 1;
    criticalsectiontwo;
    Thred2inside = 0;
    otherstufftwo;
  }
}

main()
{
  Thred1inside = 0;
  Thred2inside = 0;
  pthread_create(tid, threadone);
  pthread_create(tid, threadtwo);
}
```

Version Three

```
program versionthree;
int Thred1wantstoenter, Thred2wantstoenter;

threadone()
{
  while true do {
    Thred1wantstoenter = 1;
    while Thred2wantstoenter do {} /* busy wait */
    criticalsectionone;
    Thred1wantstoenter = 0;
    otherstufone;
  }
}

threadtwo()
{
  while true do {
    Thred2wantstoenter = 1;
    while Thred1wantstoenter do {} /* busy wait */
    criticalsectiontwo;
    Thred2wantstoenter = 0;
    otherstuftwo;
  }
}

main()
{
  Thred1wantstoenter = 0;
  Thred2wantstoenter = 0;
  pthread_create(tid, threadone);
  pthread_create(tid, threadtwo);
}
```

Version Four

```
program versionfour;
int Thred1wantstoenter, Thred2wantstoenter;

threadone()
{
  while true do {
    Thred1wantstoenter = 1;
    while Thred2wantstoenter do {
      Thred1wantstoenter = 0;
      delay(random, fewcycles);
      Thred1wantstoenter = 1;
    }
    criticalsectionone;
    Thred1wantstoenter = 0;
    otherstuffone;
  }
}

threadtwo()
{
  while true do {
    Thred2wantstoenter = 1;
    while Thred1wantstoenter do {
      Thred2wantstoenter = 0;
      delay(random, fewcycles);
      Thred2wantstoenter = 1;
    }
    criticalsectiontwo;
    Thred2wantstoenter = 0;
    otherstufftwo;
  }
}

main()
{
  Thred1wantstoenter = 0;
  Thred2wantstoenter = 0;
  pthread_create(tid, threadone);
  pthread_create(tid, threadtwo);
}
```

Dekker's Algorithm

```
program dekkersalgorithm;
int favored; /* toggle to break ties */
int T1wantstoenter, T2wantstoenter;

threadone() {
   while true do {
      T1wantstoenter = 1;
      while T2wantstoenter do {
         if favored = 2 then {
            T1wantstoenter = 0;
            while favored == 2 do {} /*busy wait */
            T1wantstoenter = 1;
         }
      }
      criticalsectionone;
      favored = 2;
      T1wantstoenter = 0;
      otherstuffone;
   }
}

threadtwo() {
   while true do {
      T2wantstoenter = 1;
      while T1wantstoenter do {
         if favored = 1 then {
            T2wantstoenter = 0;
            while favored == 1 do {} /*busy wait */
            T2wantstoenter = 1;
         }
      }
      criticalsectiontwo;
      favored = 1;
      T2wantstoenter = 0;
      otherstufftwo;
   }
}

main()
{
   T1wantstoenter = 0;
   T2wantstoenter = 0;
   favored = 1;
   pthread_create(tid, threadone);
   pthread_create(tid, threadtwo);
}
```