Too Much Milk With Locks

Both threads:

```
std::mutex mutex;

...

mutex.lock();
if (milk == 0) {
  buy_milk();
}
mutex.unlock();
```

Producer/Consumer, v1

```
class Pipe {
  Pipe() {}
  void put(char c);
  char get();

  std::mutex mutex;
  char buffer[SIZE];
  int count = 0;
  int nextPut = 0;
  int nextGet = 0;
};

void Pipe::put(char c) {
  mutex.lock();
  count++;
  buffer[nextPut] = c;
  nextPut++;
  if (nextPut == SIZE) {
    nextPut = 0;
  }
  mutex.unlock();
}
```

```
char Pipe::get() {
  char c;
  mutex.lock();
  count--;
  c = buffer[nextGet];
  nextGet++;
  if (nextGet == SIZE) {
    nextGet = 0;
  }
  mutex.unlock();
  return c;
}
```

Producer/Consumer, v2

```
class Pipe {
  Pipe() {}
  void put(char c);
  char get();

  std::mutex mutex;
  char buffer[SIZE];
  int count = 0;
  int nextPut = 0;
  int nextGet = 0;
};

void Pipe::put(char c) {
  mutex.lock();
  while (count == SIZE) {
    mutex.unlock();
    mutex.lock();
  }
  count++;
  buffer[nextPut] = c;
  nextPut++;
  if (nextPut == SIZE) {
    nextPut = 0;
  }
  mutex.unlock();
}
```

```
char Pipe::get() {
  char c;
  mutex.lock();
  while (count == 0) {
    mutex.unlock();
    mutex.lock();
  }
  count--;
  c = buffer[nextGet];
  nextGet++;
  if (nextGet == SIZE) {
    nextGet = 0;
  }
  mutex.unlock();
  return c;
}
```

Producer/Consumer, v3

```
class Pipe {
  Pipe() {}
  void put(char c);
  char get();

  std::mutex mutex;
  std::condition_variable charAdded, charRemoved;
  char buffer[SIZE];
  int count = 0;
  int nextPut = 0;
  int nextGet = 0;
};

void Pipe::put(char c) {
  mutex.lock();
  while (count == SIZE) {
    charRemoved.wait(mutex);
  }
  count++;
  buffer[nextPut] = c;
  nextPut++;
  if (nextPut == SIZE) {
    nextPut = 0;
  }
  charAdded.notify_one();
  mutex.unlock();
}
```

```
char Pipe::get() {
  char c;
  mutex.lock();
  while (count == 0) {
    charAdded.wait(mutex);
  }
  count--;
  c = buffer[nextGet];
  nextGet++;
  if (nextGet == SIZE) {
    nextGet = 0;
  }
  charRemoved.notify_one();
  mutex.unlock();
  return c;
}
```