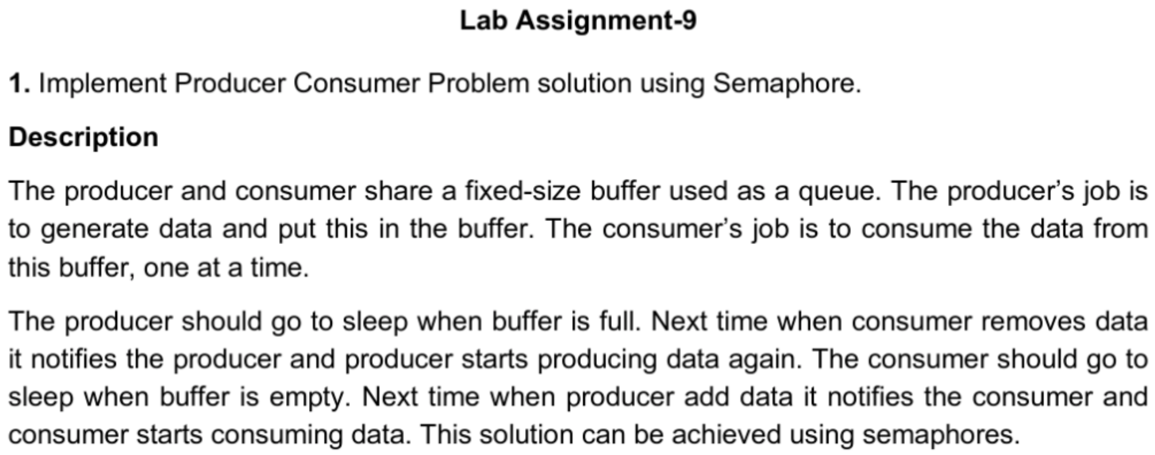
OS Lab 9

SAHIL BONDRE: U18CO021



Lab Assignment-9

1. Implement Producer Consumer Problem solution using Semaphore.

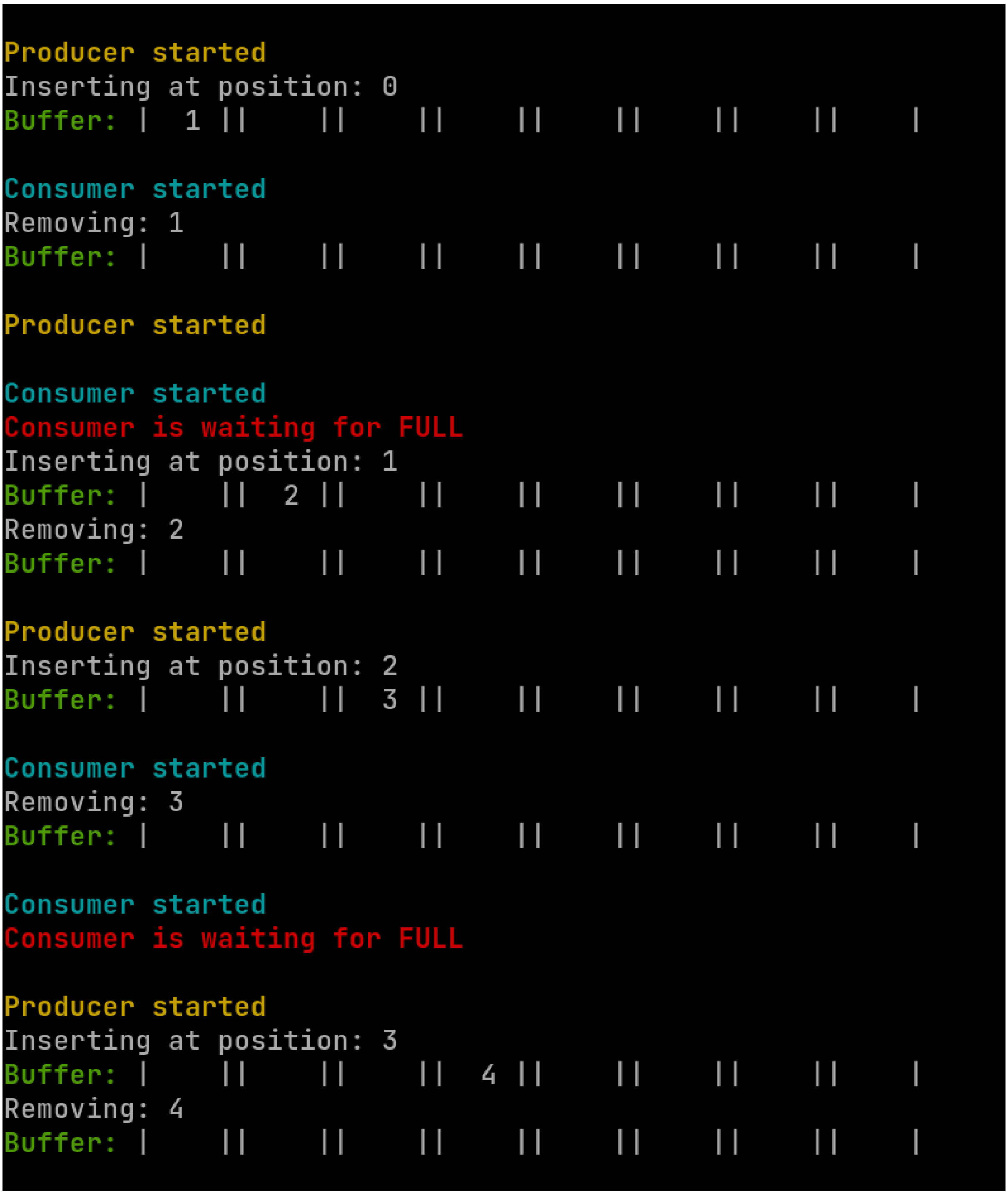
Description

The producer and consumer share a fixed-size buffer used as a queue. The producer’s job is to generate data and put this in the buffer. The consumer’s job is to consume the data from this buffer, one at a time.

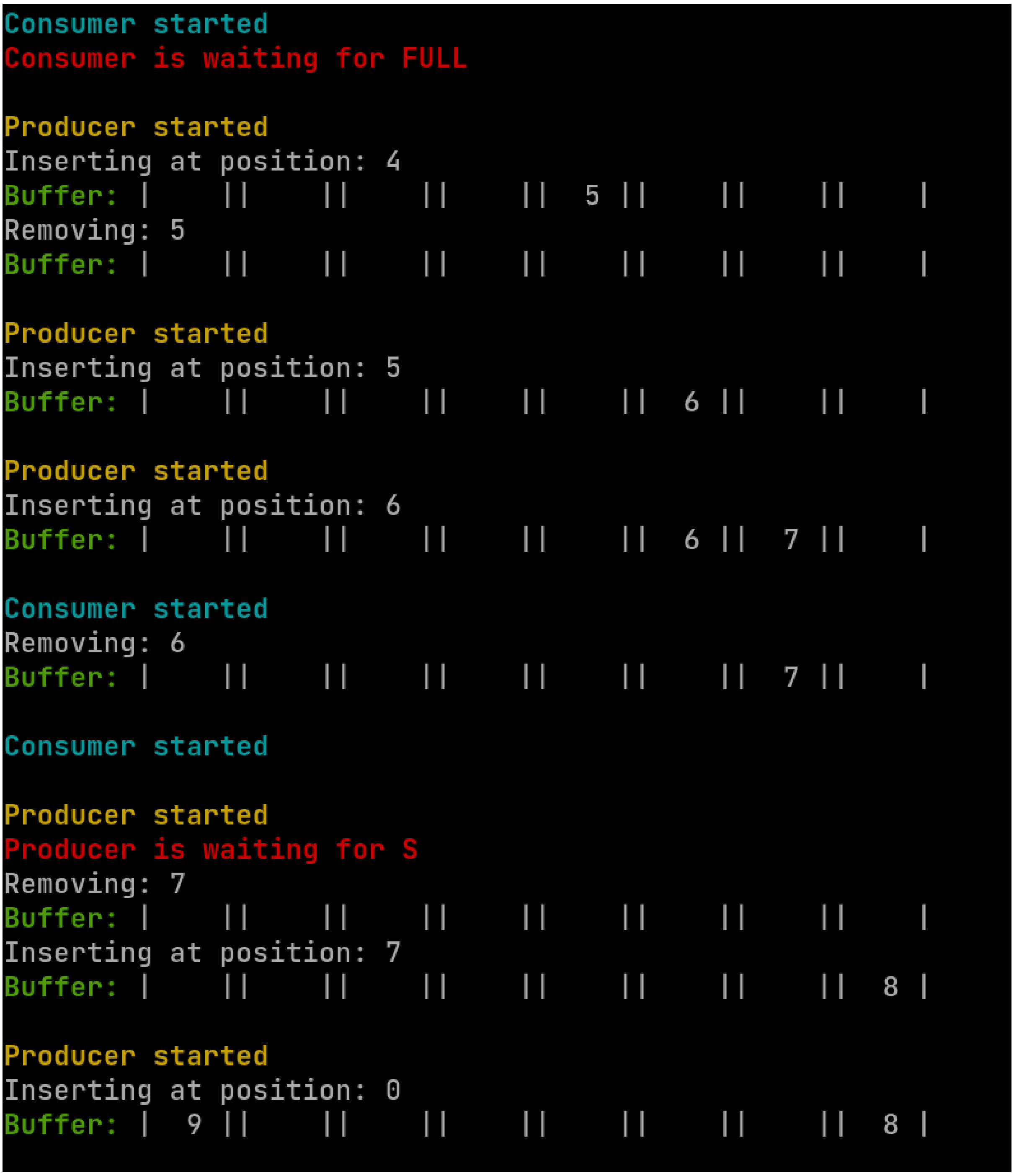
The producer should go to sleep when buffer is full. Next time when consumer removes data it notifies the producer and producer starts producing data again. The consumer should go to sleep when buffer is empty. Next time when producer add data it notifies the consumer and consumer starts consuming data. This solution can be achieved using semaphores.

main.cpp

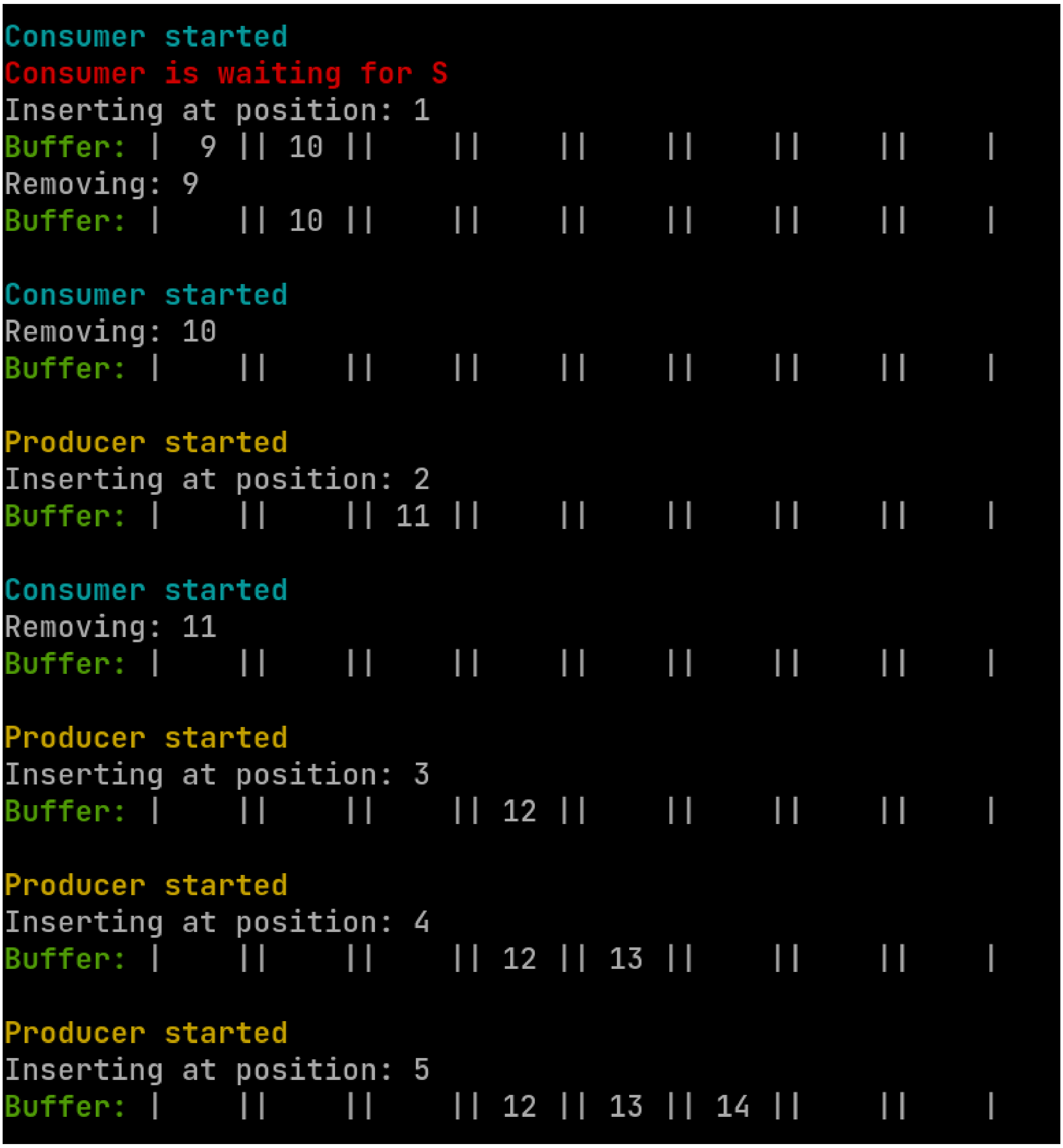
```
#include <unistd.h>

#include <functional>
#include <iostream>
#include <thread>
#include <vector>

using namespace std;

long TENTH = 100000;
long PRODUCER_DELAY = TENTH;
long CONSUMER_DELAY = TENTH;
int SIZE = 8;
int EMPTY = SIZE;
int FULL = 0;
int S = 1;
int IN = 0;
int OUT = 0;
int NEXT = 1;
vector<int> buffer(SIZE, -1);

int random_int() {
  int lowest = 0;
  int range = 10;
  return lowest + rand() % range;
}

void down_binary(int &S, string process, string chr) {
  bool waiting = false;
  while (S != 1) {
    // busy waiting
    if (!waiting) {
      cout << "\x1B[1;31m" << process << " is waiting for " << chr
           << "\033[0m\n";
    }
    waiting = true;
    usleep(TENTH);
  }
  S = 0;
}

void down(int &S, string process, string chr) {
  bool waiting = false;
  while (S <= 0) {
    // busy waiting
    if (!waiting) {
      cout << "\x1B[1;31m" << process << " is waiting for " << chr
           << "\033[0m\n";
    }
    waiting = true;
    usleep(TENTH);
  }
  S--;
}

void print_buffer() {
  cout << "\x1B[1;32mBuffer: \033[0m";
  for (auto i : buffer) {
    if (i == -1)
      printf("|    |");
    else
      printf("| %2d |", i);
  }
  cout << "\n";
}

void up(int &S) { S++; }
void up_binary(int &S) { S = 1; }

void producer() {
  while (true) {
    usleep(PRODUCER_DELAY * random_int());
    cout << "\x1B[1;33m\nProducer started \033[0m\n";
    down(EMPTY, "Producer", "EMPTY");
    down_binary(S, "Producer", "S");
    usleep(PRODUCER_DELAY);
    buffer[IN] = NEXT;
    cout << "Inserting at position: " << IN << "\n";
    print_buffer();
    ++IN;
    ++NEXT;
    IN %= SIZE;
    up_binary(S);
    up(FULL);
  }
}

void consumer() {
  while (true) {
    usleep(CONSUMER_DELAY * random_int());
    cout << "\x1B[1;36m\nConsumer started \033[0m\n";
    down(FULL, "Consumer", "FULL");
    down_binary(S, "Consumer", "S");
    usleep(CONSUMER_DELAY);
    cout << "Removing: " << buffer[OUT] << "\n";
    buffer[OUT] = -1;
    print_buffer();
    ++OUT;
    OUT %= SIZE;
    up_binary(S);
    up(EMPTY);
  }
}

int main(int argc, char const *argv[]) {
  thread t1(producer);
  thread t2(consumer);
  t1.join();
  t2.join();
  return 0;
}
```



```
Producer started
Inserting at position: 0
Buffer: |  1 ||    ||    ||    ||    ||    ||    ||    |

Consumer started
Removing: 1
Buffer: |    ||    ||    ||    ||    ||    ||    ||    |

Producer started

Consumer started
Consumer is waiting for FULL
Inserting at position: 1
Buffer: |    ||  2 ||    ||    ||    ||    ||    ||    |
Removing: 2
Buffer: |    ||    ||    ||    ||    ||    ||    ||    |

Producer started
Inserting at position: 2
Buffer: |    ||    ||  3 ||    ||    ||    ||    ||    |

Consumer started
Removing: 3
Buffer: |    ||    ||    ||    ||    ||    ||    ||    |

Consumer started
Consumer is waiting for FULL

Producer started
Inserting at position: 3
Buffer: |    ||    ||    ||  4 ||    ||    ||    ||    |
Removing: 4
Buffer: |    ||    ||    ||    ||    ||    ||    ||    |
```



```
Consumer started
Consumer is waiting for FULL

Producer started
Inserting at position: 4
Buffer: |    ||    ||    ||    ||  5 ||    ||    ||    |
Removing: 5
Buffer: |    ||    ||    ||    ||    ||    ||    ||    |

Producer started
Inserting at position: 5
Buffer: |    ||    ||    ||    ||    ||  6 ||    ||    |

Producer started
Inserting at position: 6
Buffer: |    ||    ||    ||    ||    ||  6 ||  7 ||    |

Consumer started
Removing: 6
Buffer: |    ||    ||    ||    ||    ||    ||  7 ||    |

Consumer started

Producer started
Producer is waiting for S
Removing: 7
Buffer: |    ||    ||    ||    ||    ||    ||    ||    |
Inserting at position: 7
Buffer: |    ||    ||    ||    ||    ||    ||    ||  8 |

Producer started
Inserting at position: 0
Buffer: |  9 ||    ||    ||    ||    ||    ||    ||  8 |
```



```
Consumer started
Consumer is waiting for S
Inserting at position: 1
Buffer: |  9 || 10 ||    ||    ||    ||    ||    ||    |
Removing: 9
Buffer: |    || 10 ||    ||    ||    ||    ||    ||    |

Consumer started
Removing: 10
Buffer: |    ||    ||    ||    ||    ||    ||    ||    |

Producer started
Inserting at position: 2
Buffer: |    ||    || 11 ||    ||    ||    ||    ||    |

Consumer started
Removing: 11
Buffer: |    ||    ||    ||    ||    ||    ||    ||    |

Producer started
Inserting at position: 3
Buffer: |    ||    ||    || 12 ||    ||    ||    ||    |

Producer started
Inserting at position: 4
Buffer: |    ||    ||    || 12 || 13 ||    ||    ||    |

Producer started
Inserting at position: 5
Buffer: |    ||    ||    || 12 || 13 || 14 ||    ||    |
```