Operating Systems
Constructor University
Dr. Jürgen Schönwälder

Module: CO-562
Date: 2024-10-04
Due: 2024-10-11

Problem Sheet #5

Problem 5.1: *energy drinks* (4+6 = 10 points)

Students studying late at night sometimes they run out of energy. The university decides to support these hard working students by installing vending machines offering energy drinks during the night. The machines have a certain capacity C of energy drinks they can hold and they dispense an energy drink after a student has inserted N coins. During peek hours, a student arriving at a vending machine may have to wait until it is her turn to use the machine. After dropping N coins into the machine, a student waits for the energy drink to arrive. You can assume that students always have sufficient coins. A student leaves once a drink has been dispensed.

A vending machine waits until it has received N coins and then it returns an energy drink. Once the vending machine runs out of drinks, the vending machine waits for a supplier to refill the machine to its capacity C. A supplier is servicing and refilling the vending machines every morning. Once the machine has been refilled, it provides service again to students.

a) The vending machines, the students, and the supplier are modeled by these functions:

```
void machine(void)
{
    loop {

        dispense_drink();            // dispenses a single drink

    }
}

void student(void)
{
    insert_coin();                   // inserts a single coin

    pickup_drink();                  // pickup a dispensed drink
}

void supplier(void)
{
    load_drink();                    // load a single drink

    collect_coins();                 // collect the coins
}
```

Write a solution of the synchronization problem in pseudocode using (counting) semaphores.

b) Implement the solution in C using POSIX mutexes and condition variables and no other synchronization primitives (no semaphores, barriers, message queues, ...). Make sure that only one student inserts coins and that the same student receives the drink. Other student have to wait while the machine is busy dealing with a specific student. The students and the machine also have to wait for a supplier in case the machine runs out of drinks. Finally, the machine can only start the next transaction once the previous student has picked up the drink.

Your program must support the command line option `-n` to define the number of students (with a default value of 2) and the command like option `-c` to define the capacity of the machine (with a default value of 4).

Your program should produce a trace desribing what is happening with the machine. The trace below shows how the system starts up with two students, the students getting ready for a drink, and then they wait for the machine to be refilled. Once the supplier has filled the machine, the

first student conducts a transaction and then the second student. Each line indicates how many drinks out of the capacity are currently loaded into the machine, how many coins the machine has collected, and how many coins have been inserted in the current transaction.

```
energy: [0/4 drinks, 0 coins, 0 inserted] student 0 established
energy: [0/4 drinks, 0 coins, 0 inserted] student 1 established
energy: [0/4 drinks, 0 coins, 0 inserted] supplier established
energy: [0/4 drinks, 0 coins, 0 inserted] machine booting up
energy: [0/4 drinks, 0 coins, 0 inserted] student 0 requires an energy drink
energy: [0/4 drinks, 0 coins, 0 inserted] student 0 waiting for machine to be refilled
energy: [0/4 drinks, 0 coins, 0 inserted] student 1 requires an energy drink
energy: [0/4 drinks, 0 coins, 0 inserted] student 1 waiting for machine to be refilled
energy: [0/4 drinks, 0 coins, 0 inserted] supplier arriving
energy: [0/4 drinks, 0 coins, 0 inserted] supplier loading 4 drinks
energy: [4/4 drinks, 0 coins, 0 inserted] supplier collected 0 coins
energy: [4/4 drinks, 0 coins, 0 inserted] supplier leaving
energy: [4/4 drinks, 0 coins, 0 inserted] machine waiting for more coins
energy: [4/4 drinks, 0 coins, 0 inserted] student 0 is next to be served
energy: [4/4 drinks, 0 coins, 1 inserted] student 0 inserted another coin
energy: [4/4 drinks, 0 coins, 2 inserted] student 0 inserted another coin
energy: [4/4 drinks, 0 coins, 3 inserted] student 0 inserted another coin
energy: [4/4 drinks, 0 coins, 4 inserted] student 0 inserted another coin
energy: [4/4 drinks, 0 coins, 5 inserted] student 0 inserted another coin
energy: [4/4 drinks, 0 coins, 5 inserted] student 0 waiting for drink to arrive
energy: [3/4 drinks, 5 coins, 0 inserted] machine dispensing drink
energy: [3/4 drinks, 5 coins, 0 inserted] machine waiting for pickup of drink
energy: [3/4 drinks, 5 coins, 0 inserted] student 0 picked up a drink
energy: [3/4 drinks, 5 coins, 0 inserted] student 0 enjoying an energy drink
energy: [3/4 drinks, 5 coins, 0 inserted] student 1 waiting for the machine
energy: [3/4 drinks, 5 coins, 0 inserted] machine waiting for more coins
energy: [3/4 drinks, 5 coins, 0 inserted] student 1 is next to be served
energy: [3/4 drinks, 5 coins, 1 inserted] student 1 inserted another coin
energy: [3/4 drinks, 5 coins, 2 inserted] student 1 inserted another coin
energy: [3/4 drinks, 5 coins, 3 inserted] student 1 inserted another coin
energy: [3/4 drinks, 5 coins, 4 inserted] student 1 inserted another coin
energy: [3/4 drinks, 5 coins, 5 inserted] student 1 inserted another coin
energy: [3/4 drinks, 5 coins, 5 inserted] student 1 waiting for drink to arrive
energy: [2/4 drinks, 10 coins, 0 inserted] machine dispensing drink
energy: [2/4 drinks, 10 coins, 0 inserted] machine waiting for pickup of drink
energy: [2/4 drinks, 10 coins, 0 inserted] student 1 picked up a drink
energy: [2/4 drinks, 10 coins, 0 inserted] student 1 enjoying an energy drink
energy: [2/4 drinks, 10 coins, 0 inserted] machine waiting for more coins
```