```
#DEFINE C = 4 // Drink Capacity
#DEFINE N = 5 // Drink Price in Coins


// Initialize semaphores
semaphore access = 1          // Binary semaphore to limit access to one
person at a time
semaphore supplier_call = 0      // Semaphore for supplier call
semaphore drinks = 0     // Semaphore for the number of available drinks

// Shared variables
int capacity = C;             // Total capacity of the vending machine
int coins_needed = N;         // Number of coins needed for a drink

// Vending Machine Process
void machine(void) {
    while (true) {
        if (drinks==0){
            signal(supplier_call); // Notify supplier if no drinks left
        }
        wait(drinks); // Wait for drinks to be full
        dispense_drink(); // Dispenses a single drink
    }
}

// Student Process
void student(void) {
    while (true) {
        wait(access); // waiting for access
        for (int i =0;i<coins_needed;i++){
            insert_coin(); // keep adding coins until we insert enough
        }

        signal(drinks) // Signaling drinks that a drink was removed
        pickup_drink(); // Pickup a dispensed drink
        signal(access); // giving up access
    }
}

// Supplier Process
```

```
void supplier(void) {
    while (true) {
        wait(supplier_call); // wait to be called and have access
        signal(access); // Take access
        for (int i=0;i<capacity;i++){
            load_drink(); // Load a single drink
        }
        collect_coins(); // Collect coins after refilling
        signal(supplier_call); // turning off call
        signal(access); // giving up access
    }
}
```