Design and pseduocode

List of semaphores used:

Semaphore max\_capacity = 50 // for max number of customer

Semaphore service\_counter = 3 // for number of available postal worker

Semaphore cust\_ready = 0 // for a customer to be ready to serve

ArrayList<semaphore> service\_finished // separate semaphore for each customer with different services

Semaphore leave\_service\_counter = 0 // for customer has received service and can leave post office

Semaphore mutex\_inner = 3 // postal worker cannot access customer information until customer has safely written to it

Semaphore mutex\_outer = 1 // for limit other customers from writing to global customer data until a postal worker has read this customer data

Semaphore mutex\_greet = 0 // let customer know the postal worker has successfully read customer data

Pseudocode

Customer thread:

Void customer()

{

semWait(Max\_capacity);

semWait(service\_counter);

//customer enter post office;

semWait(mutex\_outer);

semWait(mutex\_inner);

//get customer number and customer service;

semSignal(cust\_ready);

semSignal(mutex\_inner);

semWait(mutex\_greet) ;

//customer finish service;

semSignal(mutex\_outer);

semWait(service\_finished);

//wait until assigned postal worker completed the service

semSigal(leave\_service\_counter);

//customer leave post office;

semsignal(max\_capacity);

}

Postal worker thread:

Void postalworker

{

semWait(cust\_ready);

semWait(mutex\_inner);

//fetch customer number and service;

//postal worker service customer;

semSignal(mutex\_greet);

semSignal(mutex\_innter);

//Time based on different service;

//Postal worker finished serving customer;

semSignal(service\_finished);

semWait(Leave\_service\_counter);

semSignal(service\_counter)

}

Postoffice

{

//initialize different semaphore

//print simulating post office with customer and postal workers

//create customer objects and threads

for( number of customers..);

//create postal worker objects and threads

for( number of postal workers … );

// join customer thread back

for( number of customers … );

exit

}