Mason Godfrey (mgodfrey@csu.fullerton.edu)

Bijaya Shrestha (sthavjay@csu.fullerton.edu)

CPSC 474

Project 1

PSEUDOCODE FOR OUR PROGRAM

1) Pseudocode for main function:

   a) Create an event sequence with a string form in a matrix form (row and column)
   b) Create a logical sequence with all int variables in a matrix form. (row and column)
   c) Print out the calculation as depends with output of each header files

2) Pseudocode for header.h file.
   a) Forming a dimension of the matrix which is 3*4 matrix.

3) Pseudocode for algorithmVerify header file for problem 2 and 3.
   a) Using an int *algorithmnVerify* function for the storing the event sequence after verifying the matrix is possible or not.
   b) Using if statement inside While loop for determining the max number and assign the max until it hit to end of the sequence.
   c) If any number of sequences is not equal to max, then the output prints "incorrect" for the matrix
   d) At first, initialize every array of the matrix null.

e) Create a loop and if statement to know the arrays are in increasing order and initialize ‘r’ as a receiver if the current number is greater than previous plus 1.
f) Check and find the sender which must be less than exactly 1 and mark it as “s”.
g) Use an if statement for checking a sender and also check the whether it is passing through null or not.
h) Once the check is done, set the current array into letter and move to next array and finish the loop until to the end.
i) And print out the logic clock.

4) Pseudocode for algorithmcalculate header file for problem 1.

a) *int AlgorithmCalculate(string eventSequence[row][col]):* Creating an algorithmncalculate function to storing all the logic clock as well as output of the calculation.
b) Initialize k as the value immediately before a in the same row
c) Initialize logicClock to store our logic clock
d) Initialize logicClocks to 0, Initialize responsePts to -1
e) create a boolean for final result that matches the output with logicclock.
f) Create a while and for function to travel each row and colum respectively and update the logic clock.
g) If we're not a response point or going past a response point, update our logic clock.

h) For all rows, check if and where there is a response point, and update the logic clock if we find a response point before resetting it
i) Set k equal to the current clock # in the column which will become the previous item.
j) Set k to 0 for a new row.
k) Check if we're done with the logicclock by knowing Boolean done = false or true.
l) And Printout the logicClock