CE464 Lab 8 – Coding Assignment

Code: Implementing a Library for Your Program #3 Window Management

Lab Coding Assignment – Window Management - In this lab, you are to implement a Library that will manage your window data structure on server and rcopy. As it states in program #3, you must implement a library (so a .c and .h file) to manage your windowing.

From the programming assignment on the **Window Library:**

> You **must** implement your windowing functionality as a library. This means that the window data structures, and accessor functions must be defined in a single c (c++) file (and single .h file) separate from your other code. Both your client (rcopy) and server must use this code (your library) for windowing. You are allowed to have up to (at most) 3 global variables in this file, but these global variables cannot be accessed by code in any other file. The only code that may be in this C/C++ file is your windowing code. All access to your windowing data must be via your window library's accessor functions.
>
> Note – you must implement your windowing functionality and cannot use a preexisting data structure (e.g. cannot use something provide by C++). Your window data structure must be a malloc()ed array of structures and managed as a circular queue (e.g. index into window = packet-sequence-number % window size).

The window management functionality needed by the server and are rcopy are different. For rcopy, its really just buffer management, for the server its sliding window management using Selective-Reject ARQ. So, while there is some overlap, you will probably want to have some functions that are specific to the server and some specific to rcopy. I recommend you not force a function to work for both.

I. Server Window management

1. You need provide these print function to use in the testing your windowing functionality. These functions should be part of your library.
   a) Function to print out your Server's Window meta data:

```
Server Window - Window Size: 4, lower: 0, Upper: 4, Current: 0
```

   b) Function to print your entire window (from 0 to window size -1). For each valid entry print out its Sequence Number and PDU size. If the entry is not valid (so nothing entered into the window entry) print "not valid").

```
Window size is: 4
   0 sequenceNumber: 16 pduSize: 12
   1 sequenceNumber: 17 pduSize: 13
   2 not valid
   3 sequenceNumber: 15 pduSize: 11
```

2. **Server window test cases** (write a test program that uses your library[1]):

   a. Simple test: Just adding PDUs to your Server window and processing a SREJ
      i. Window size of 4
      ii. Loop through and add PDUs with sequence numbers 0-3
      iii. Print out your window
      iv. Print out your window meta data (current, upper, lower)
      v. Selective Reject test: process a SREJ of the PDU with sequence number 2 and print this PDU.
      vi. **Sample output**:

```
husmith:srejWindow$ windowTest
SREJ testing (srej 2)
Window size is: 4
   0 sequenceNumber: 0 pduSize: 32
   1 sequenceNumber: 1 pduSize: 25
   2 sequenceNumber: 2 pduSize: 35
   3 sequenceNumber: 3 pduSize: 22
Server Window - Window Size: 4, lower: 0, Upper: 4, Current: 4 window open?: 0

Pdu from window: Seq Num: 2 pduSize: 35
```

Print out the pdu (with seq#2) info that was retrieve from the window buffer

   b. Implement the following pseudo code to interact with your library. This test code must be in a file separate from your library.

```
3   While(1)
4
5       if your window is open
6           create a pdu and add it to your window
7           print out your window meta data
8       else when your window is closed
9           print your window
10          print your window meta data
11          scanf a sequence number to RR
12          RR the sequence number
13          Print your window
14          Print your windows meta data
15
16  end while
```

(see figure on next page for sample output)

[1] The test code (and its main()) must be in a file separate from your library.

Sample output based on this pseudo code:

Print out the window meta data as the PDUs are added

```
husmith:srejWindow$ windowTest
Server Window - Window Size: 4, lower: 0, Upper: 4, Current: 0 window open?: 1
Server Window - Window Size: 4, lower: 0, Upper: 4, Current: 1 window open?: 1
Server Window - Window Size: 4, lower: 0, Upper: 4, Current: 2 window open?: 1
Server Window - Window Size: 4, lower: 0, Upper: 4, Current: 3 window open?: 1
Window size is: 4
   0 sequenceNumber: 0 pduSize: 32
   1 sequenceNumber: 1 pduSize: 25
   2 sequenceNumber: 2 pduSize: 35
   3 sequenceNumber: 3 pduSize: 22
Server Window - Window Size: 4, lower: 0, Upper: 4, Current: 4 window open?: 0
```

Window is full (print out the entire window)

```
Enter number to RR: 3
Processing RR
Window size is: 4
   0 not valid
   1 not valid
   2 not valid
   3 sequenceNumber: 3 pduSize: 22
Server Window - Window Size: 4, lower: 3, Upper: 7, Current: 4 window open?: 1
Server Window - Window Size: 4, lower: 3, Upper: 7, Current: 5 window open?: 1
Server Window - Window Size: 4, lower: 3, Upper: 7, Current: 6 window open?: 1
Window size is: 4
   0 sequenceNumber: 4 pduSize: 35
   1 sequenceNumber: 5 pduSize: 34
   2 sequenceNumber: 6 pduSize: 32
   3 sequenceNumber: 3 pduSize: 22
Server Window - Window Size: 4, lower: 3, Upper: 7, Current: 7 window open?: 0
```

RR: 3 (so PDUs with Seq# 0, 1 , 2 are no longer in the window) Print out window.

Then create 3 new PDUs

Window is full again (after adding 3 more PDUs) Print out window

```
Enter number to RR: 5
Processing RR
Window size is: 4
   0 not valid
   1 sequenceNumber: 5 pduSize: 34
   2 sequenceNumber: 6 pduSize: 32
   3 not valid
Server Window - Window Size: 4, lower: 5, Upper: 9, Current: 7 window open?: 1
Server Window - Window Size: 4, lower: 5, Upper: 9, Current: 8 window open?: 1
Window size is: 4
   0 sequenceNumber: 8 pduSize: 24
   1 sequenceNumber: 5 pduSize: 34
   2 sequenceNumber: 6 pduSize: 32
   3 sequenceNumber: 7 pduSize: 28
Server Window - Window Size: 4, lower: 5, Upper: 9, Current: 9 window open?: 0
```

RR: 5 (so PDUs with Seq# 3 and 4 are no longer valid in the window) Print out window. Then create 2 new PDUs

Window is full again (after adding 2 more PDUs)