COMP 322/L Introduction to Operating Systems and System Architecture
Assignment #2—Solving the Producer-Consumer Problem with Semaphores

Objective:

To implement the solution to the Producer-Consumer problem, using P and V operations, and prevent over-producing and/or over-consuming of data in a shared circular buffer.

Specification:

The program arbitrarily fills (produces) data in a shared circular buffer and removes (consumes) the data. A menu controls the operations, and each choice calls the appropriate procedure, where the choices are:

1) Enter parameters
2) Simulate on a shared circular buffer
3) Quit program and free memory

Assignment:

- The buffer is a dynamic array of n integers, each indexed value initialized to 0
- The producer and the consumer are represented by separate functions, chosen to execute by a binary random selection (eg. 0=producer runs, 1=consumer runs)
- The producer fills the buffer by adding a 1 to the next *k1* slots of the buffer, modulo *n*, where *k1* is a random number between 1 and *maxfillsize*.
- The consumer empties the buffer by resetting to 0 the next *k2* slots of the buffer, modulo *n*, where *k2* is a random number between 1 and *maxemptysize*.
- The P (decrement) and V (increment) operations are implemented on sempahores *e* and *f* to test for an empty (*f*=0) or a full (*e*=0) buffer to prevent the producer from over-producing and the consumer from over-consuming.

What NOT to do (any violation will result in an automatic score of 0 on the assignment):

- Do NOT modify the choice values (1,2,3) or input characters and then try to convert them to integers--the test script used for grading your assignment will not work correctly.
- Do NOT turn in an alternative version of the assignment downloaded from the Internet (coursehero, chegg, reddit, github, etc.)
- Do NOT use any self-created or external libraries that cannot be located/utilized by zylabs
- Do NOT turn in your assignment coded in another programming language (C++, C#, Java, Python, Perl, etc.)—it will NOT compile under zyLabs C compiler.

What to turn in:

The source code as a C file (**comp322_asmt2.c**) uploaded to zyLabs by the deadline of 11:59pm PST (-20% per consecutive day for late submissions, up to the 4th day—note 1 minute late counts as a day late, 1 day and 1 minute late counts as 2 days late, etc.)

Sample test run (Inputs: 1 20 5 5 10 2 3):

(Note: C[5] signifies that the consumer tried to empty 5 cells, while P[1] signifies that the producer tried to fill 1 cell)

```
Producer-Consumer problem
-------------------------
1) Enter parameters
2) Simulate on a shared circular buffer
3) Quit program and free memory

Enter selection: 1
Enter size of the buffer: 20
Enter maximum fill size: 5
Enter maximum empty size: 5
Enter maximum number of iterations: 10

Producer-Consumer problem
-------------------------
1) Enter parameters
2) Simulate on a shared circular buffer
3) Quit program and free memory

Enter selection: 2
C[5]: Buffer=00000000000000000000
C[4]: Buffer=00000000000000000000
P[5]: Buffer=11111000000000000000
P[1]: Buffer=11111100000000000000
P[4]: Buffer=11111111110000000000
C[2]: Buffer=00111111110000000000
P[5]: Buffer=00111111111111100000
P[5]: Buffer=00111111111111111111
C[5]: Buffer=00000001111111111111
P[1]: Buffer=10000001111111111111

Producer-Consumer problem
-------------------------
1) Enter parameters
2) Simulate on a shared circular buffer
3) Quit program and free memory

Enter selection: 3
Quitting program
```