CSCE 4600, Fall 2020
Project #1
Due: 3-22-2020 on Canvas

For this project you will be working in groups of four. Each member of your group **must** sign up on the corresponding project sheet in order to receive a grade. Your task is to **implement** a semaphore-based solution to the problem stated below.

Consider a system with N blocks of storage, each of which holds one unit of information (e.g. an integer, character, or employee record). Initially, these blocks are empty and are linked onto a list called *freelist*. Three threads communicate using shared memory/global variables in the following manner:

Shared Variables: freelist, list-1, list-2: block (where block is some data type to hold items)

```
Thread-1
var b: pointer to type block;
while (1)
{
  b:= unlink(freelist);
  produce_information_in_block(b);
  link(b, list1);
}
```

```
Thread-2
var x,y: pointer to type block;
while (1)
{
  x:=unlink(list-1);
  y:=unlink(freelist);
  use_block_x_to_produce info_in_y(x, y);
  link(x, freelist);
  link(y, list-2);
}
```

```
Thread-3
var c: pointer to type block;
while(1)
{
  c:=unlink(list-2);
  consume_information_in_block(c);
  link(c, freelist);
}
```

Using the POSIX library, rewrite the code for the threads, using semaphores to implement the necessary mutual exclusion and synchronization. The solution must be deadlock-free and concurrency should not be unnecessarily restricted.

Deliverables: You must submit a working solution that compiles without errors on the CSE computers. Further, you must submit a detailed description of the approach you have used to solve the problem.

Grading: Your project will be graded on program correctness, particularly whether it is avoiding deadlocks and its ability to maximize concurrency among the three processes. Further, the detailed description of your approach and the synchronization mechanisms used will weigh in the final project grade.

Only one member of each group should submit on behalf of the entire group.