CSc 360: Operating Systems (Spring 2022)

Programming Assignment 2: Deliverable A

Alex Holland

1.
The number of threads created will be equal to the number of trains in the program. Thus, the number of threads will be at least equal to the number of rows in the input file. In addition there will be the main thread.

2.
The threads will work independently. The program will not contain an overall controller thread.

3.
The program will use the following two mutexes:

- A mutex will be created so that it can lock the station queues. If two trains finish loading at the same time only one train will be allowed to access the queue.
- A mutex will be created so that when a train is crossing the track, no other train will be granted access to cross until the train on the track has completely crossed.

4.
The main thread will not be idle. The main thread will be used to schedule the and dispatch the trains.

5.
The program will use four queues to represent stations. The queues will implemented via singly linked lists. In addition, the program will use a typedef struct to store the train number, priority, direction, loading time, and crossing time.

6.
To ensure that data structures in the program will not be modified concurrently the use of of mutexes (*pthread_mutex_lock* & *pthread_mutex_unlock*) will be used to protect shared resources from the various train threads in the program.

7.
The program will use two condition variables.

(a)
The first condition variable (convar1) will be used to represent when a train has finished it's loading process. The second condition variable (convar2) will be used to represent the condition of the track, it will be used to track whether the track is currently being used by a train or not.

(b)
The first mutex will lock the station queues, this mutex will be associated with the first condition variable (convar1) that represents when a train has finished it's loading process. Another mutex will be used to

protect the track from being used with more then one train at a time, this mutex will be associated with the the track condition variable (convar2).

(c)
After *pthread_cond_wait*() has been unblocked and re-acquired the mutex the following should happen. With respect to the station queue condition variable; after a train has been loaded it will be put into a queue. With the main track condition variable, the main thread will dispatch the highest priority train onto the main track.

8.

```
Open and read the input file line by line
Store the information of each line (train) into an array of structs
Create a thread for each train
Broadcast to the trains to begin loading and start a timer
While there are still trains that have not crossed the track:
    Call a priority function to determine which train needs to cross first
    Get the station mutex, put the train into the queue, then release station mutex
    Declare to the dispatcher that a train is ready to cross the track
    Get the track mutex, utilize the main track, then release the track mutex
    Now that the track is clear, the next highest priority (loaded) train can cross
At the end of the program destroy the train threads and exit the program
```