**PROJECT 1**

For part one of this assignment, we had to design the data structures and back end functions for the airport simulation. That is, we had to implement the basic event engine, construct the arrival loop, and code definitions for all the major events and data structures.

After that, we designed a priority queue, priority.c, to handle events. This queue was designed as a heap array, heapified so that the smallest time value is at the top of the heap. The functions created in priority.c are then used in event.c. This event.c file is a wrapper for the priority functions, used to created and manage the events passed in and out of the queue. The file randsim.c is used to generate the times at which events occur, which are called inside the time.c file. Only the arrival time portion of the time.c file functions at this time.

Main.c performs the function calls to create and manage the simulation of these events. This is done by creating an initial event, then passing it into a while loop with a switch statement that performs the correct action based on the event type attached to the event. This process is repeated until the desired number of passengers is reached, at which point the remaining events are resolved.

Priority.c is based on code that we found online, modified to fit our needs and to be clearer for our understanding of it. This file contains functions that allow us to initialize the heap to the desired maximum size, insert an event at the bottom of the heap, remove an event from the top of the heap, and finalize the heap. There are also functions to check if the heap is empty or full. To heapify the heap, we used two functions. One, bottom to top, for when we insert a value to the heap, and a top to bottom function for when we pop an event off the heap. Each of these heapifies the heap from the point of the change we made to ensure the heap retains it properties through inserts and removes.

Event.c is a wrapper for priority.c, so most of its code is function calls for the priority queue. It includes modifications to certain parts of the code to allow it to work with the passenger and queue data structures. The finalize is modified to clear the heap, the queue, the event, and the passenger related to the event. The event\_create is unique to event.c. This function creates a new event, and the passenger and queue associated with it. Destroy removes an event from the queue and frees it. Event\_schedule adds an event to the priority queue. Event\_cause pops off the top event from the heap. It is used to cause an event to happen in the simulation.

Queue.c is the FIFO queue that will be used in the later parts of the project. It is partially implemented, though it is not used in this part of the project, therefore, we simply made sure the barebones of it exist and it does not cause the program to fail to compile.

Randsim.c was given to us, and we made no changes, simply implementing some of its functions where needed.

Time.c is again partially implemented for this part. At this time, you can get and set times, as well as call the arrival and airline queue times. The stubs for the functions to be used later currently exist but are not fully implemented in this version of the product.

We also created an additional file called defs.h that contains all the necessary includes to link the files and run the program. We decided to include all of the header files and library files into a defs.h file to reduce redundancy, and the order in while the program files are included is specific to the order in which the program needs them in order to function properly. This file is included into each “.c” file with an ifndef statement that checks if it has already been included for the file every time the program has to access the file.