allen_take_off_time

Marissa Allen

Overview:

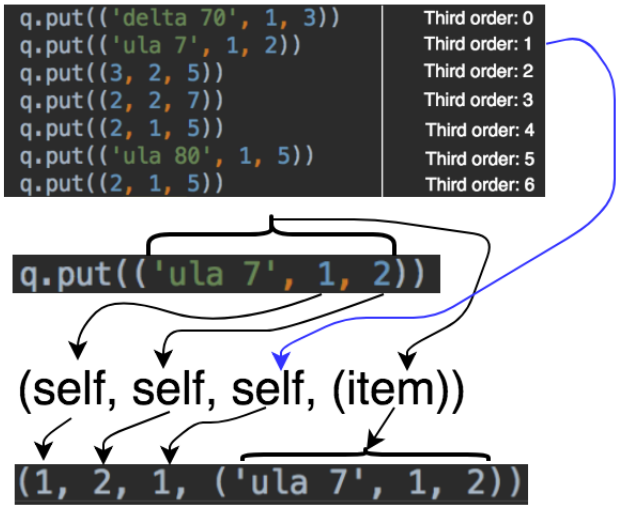
The purpose of this project is to keep track of the information needed to schedule the airstrip resource for a queue of airplanes. As soon as airplane rolls out onto the taxiway they radio in their request to ground control and say that want to take off at a certain time slot. These requests are held in a file that is submitted to the allen_take_off_time program. This program will take care of the request identifier, request submission time, time slot requested, length of time requested, actual start time, and actual end time. It will do this by creating a class for all of the request information and combining it with a priority queue to sort the information. As the planes move along in their time slots the status of the queue will be printed out and a list will be printed of the plane's actual take off times.

Design:

This program contains four modules - allen_take_off_time, priority_queue, airstrip_schedule, and test_allen_take_off_time. The test_allen_take_off_time module is a unit test file to test any edge cases. For this program, the allen_take_off_time module will have a function called file_read to take in input from a file, read the request lines, separate the strings into a list containing one line per element, and load the file requests into the constructor of the class in the airstrip_schedule module. The file request information that the module will keep

track of is the request identifier, request submission time, time slot requested, length of time requested, actual start time, actual end time. It will then load all of this information into a priority queue using the built-in priority queue module. As time moves forward the printed status of the queue will show the planes waiting for take off that has been created as a toString( ) from the airstrip_schedule module. Then the program will write the actual listing of take off times to the command line for the user.

A diagram and explanation have been included to further breakdown how the priority queue works. The priority queue is a list that has tuples inside of it. The priority queue is ordered by (self, item) and it has three tuple elements in front of the item. Which makes a four-tuple inside of a queue displayed below as (self, self, self, (item)). The second element of the entered



item data is set as the highest priority and placed in the first position of the four-tuple. The third element of the item data is set as the next highest priority and placed in the second position of the four-tuple. The third position of the four-tuple is derived from the order that the item is placed in using the put function. This addition also avoids conflicts with pilots that have the same

scheduling time requests by recording the order of the request. If two of the same request were placed into the queue at the same time, the plane that first put in the request would still have priority over the plane that put in their request after them. This element numbers each tuple as it is placed into the queue from zero to the length of the queue minus one. Now when the plane requests are taken from the queue, they are ordered by the first three elements of the four-tuple, and the item data is displayed to the user.