Lab 04 Process Queue Simulator

Bradley Grose

# Problem

In this lab, we must make a program that creates a first come first serve scheduling using a queue. We are given a driver as the main method of the code. We must create a class Process that takes in a name and a completion time and has the object methods. Then we have a class LinkedListQueue that must make a generic linked list that has queue properties. Finally, we have a class ProcessScheduler that has the linked list as well as runs the current processes, adds processes, runs the next processes, and cancels and prints processes.

# Solution

Starting with the driver, I inserted the driver file into my project as ProcessSchedulerSimulator. Then I created the object Process. In this object there are a String for the name and a double for the completion time. I have a default and a parameterized constructor, as well as accessors and mutators to check for correct values, as time must be greater than 0. Finally, I have a to string function to return to driver. Then I created the object LinkedListQueue of type T. First, I create a List node that has the data and link values. Then there are a head, tail, and size variable that are set. It creates a peek which gives the head of the list, as well as an enqueue which loads a Node to the top of the head, as well as a dequeue function that removes the node from the head. Finally, there is a print data function that takes the head and sends it to string and then keeps going down the list using a temp link to the end of the list. There is also a ProcessScheduler class that creates a linked list of type process. It has a add Process function that adds to the queue a process, as well as a runNextProcess that goes to the next queue in the list. There is also a cancelProcess that does the same thing. Finally, there is a print function that calls the linked list print functions.

# Implementation Problems Encountered

No problems were encountered in this lab.

# Lab Report Questions

1. A queue is structed as a linked list or array that holds data and can be manipulated on a first in first out methodology to access the data.
2. A stack has all inserts and deletions made on one end of the list or array while a queue uses one end to insert data and the other end to remove data.