# Project 3

Joshua Shor, Tim Nguyen

Submitted Files

**Makefile:** A file that allows the user to compile all the files listed below and run the

**README.md:** A readme file that just explains a bit about the program

**pcb.h:** The header file that instantiates the classes for the process control block

**scheduler.h:** The header file for the base scheduler class

**scheduler_fcfs.h:** Header file that has class SchedulerFCFS to implement the FCFS Algorithm

**scheduler_priority.h:** Header file that initializes the SchedulerPriority class

**scheduler_priority_rr.h:** Header file that initializes the SchedulerPriority class

**scheduler_rr.h:** Header file that initializes the SchedulerRR class

**scheduler_sjf.h:** Header file that initializes the SchedulerSJF class

**scheduler_fcfs.cpp:** this program file contains the FCFS Algorithm

**scheduler_priority.cpp:** Priority scheduling algorithm file which contains all the functions needed.

**scheduler_priority_rr.cpp:** Scheduler class implements the RoundRobin (RR) scheduling algorithm

**scheduler_rr.cpp:** Scheduler class implements the RoundRobin (RR) scheduling algorithm.

**scheduler_sjf.cpp:** Scheduler class implements the SJF scheduling algorithm.

**driver_fcfs.cpp:** driver program for the FCFS algorithm

**driver_priority.cpp:** driver program for the priority algorithm

**driver_priority_rr.cpp:** driver program for the priority round robin algorithm

**driver_rr.cpp:** driver program for the round robin algorithm

**driver_sjf.cpp:** driver program for the shortest job first algorithm

**To Compile and Run the program:**

1. First have the directory in the terminal be the same directory with all the program files

2. Type make to compile the program
3. The make file will create a compiled program a.out, type in ./a.out to run the program

**Results**

The program ran well for all scheduler algorithms, we didn't encounter any issues when we ran the code. All the outputs ran as expected and we didn't have any problems with any of the outputs. Everything went well as we expected.

**Program functions:**

*simulate():* this function is critical and present in all the scheduler algorithm files. Depending on which method of scheduling was used, the function would be written differently to account for whatever scheduler is used.

*SchedulerSJF():* a constructor used to implement the Shortest Job First algorithm

*SchedulerFCFS():* a constructor used to implement the First Come First Serve algorithm

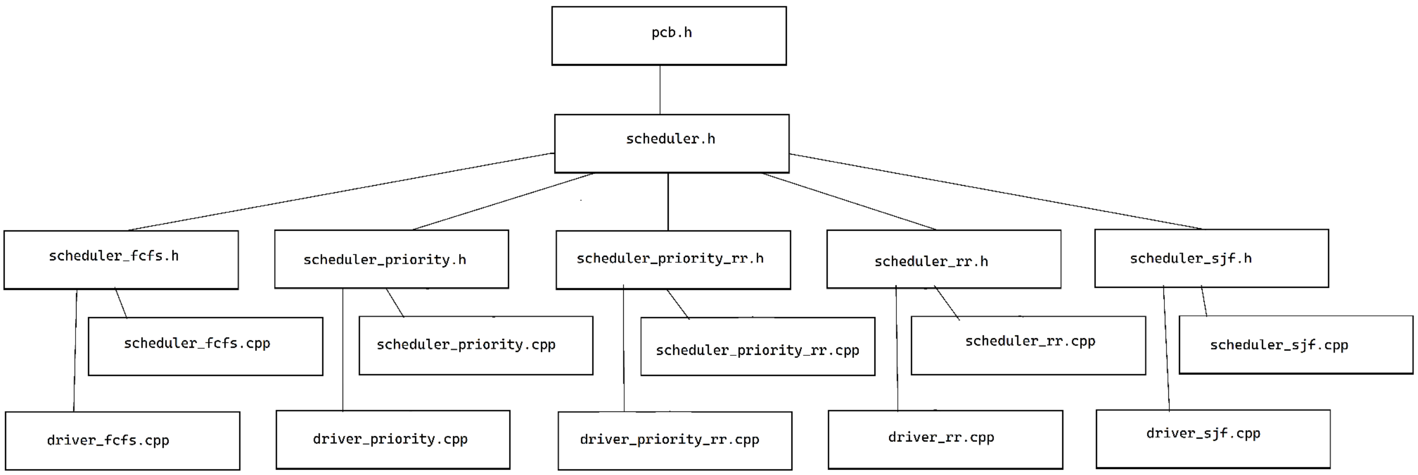
*SchedulerPriority():* a constructor used to implement the Priority algorithm

*SchedulerRR():* a constructor used to implement the Round Robin algorithm.

*Scheduler():* The base scheduler in which the above functions inherit the variables and other functions

*PCB():* the functions used to create the process control blocks

**Implementation:**



In the beginning of the program we added the aforementioned functions in our program that would be called by the related driver functions. PCB is the highest parent classes that would dictate all the classes in the other child classes. PCB inherits to scheduler, and then The functions within Scheduler would be inherited by the child classes: scheduler_fcfs, scheduler_priority, scheduler_priority_rr, scheduler_rr, and scheduler_sjf.those classes. (see picture above)

In PCB, we had a PCB() function that constructed a new Process Control Block object each time as needed. Then we called a print function to print out of the process along with its priority and burst time. We also compared each burst from the last. Scheduler is the base class for all the scheduling algorithms that were implemented further. Each scheduler algorithm file had their own variables for their respective functions and classes. Common to each scheduler header and c++ files were the constructors, the initializers which initialized the PCB vector, print_results() which would print out the results of the program while running, and the simulate() function which is where the scheduling algorithm takes place. The initialize function in each algorithm takes in the vector PCB, and then dumps them into another vector called schedulerList in whatever order they need to be. For example, the shortest job first will put in each process ordered by the shortest one to the longest one inside the schedulerList vector. In simulate, the schedulerList vector is processed. Inside either a for loop or while loop,  first turnaround time is calculated, and then wait time is then calculated afterwards. Average wait time and average turn around time is incremented by each time the loop is running. Finally at the end of the loop, it pops off the top element from schedulerList and then runs the next element inside the vector.

**Future Improvements:**

One of the things we could improve upon is to make sure the program is able to handle a larger amount of processes for scheduling and can run for longer periods of time. Maybe another improvement would be to have the program check to see which scheduler to use for various different processes and pick which one would work the best. Otherwise the program ran fine with no errors. cppcheck test displayed all clear for our program.