**Programming Assignment One**

Finish all the tasks below. Submit the code for all questions, that is, submit three ***.c*** programs. Add the proper descriptions as the comments at the beginning of the code, including author, student ID, function description, and create date. Refer to Rubrics for the grading criteria.

The goal of this homework assignment is to implement the **Round Robin CPU (RR) scheduling algorithm** and **Shorest Job First (SJF) algorithm**.

**Task 1: SJF**

Write a program that does the following.

1. Write a program called **sjf.c**. This program asks the user to enter the number of processes to schedule (you can assume that this number is always positive, there is no need to check for that in your program).
2. For each process to schedule, ask the user to enter the burst time for the process (you can assume that all burst times are positive, there is no need to check for that in your program).

Here is an example of what running the program must look like (where **4**, **6**, **3**, **1**, and **7** are inputs typed by the user on the keyboard):

============================================================

Enter the number of processes to schedule: **4**

Enter the burst time of process 1: **6**

Enter the burst time of process 2: **3**

Enter the burst time of process 3: **1**

Enter the burst time of process 4: **7**

Schedule sequence: 3 2 1 4

AWT: 3.750000

ATT: 8.000000

============================================================

**Task 2: RR**

Write a program that does the following.

1. Write a program called **rr.c**. This program asks the user to enter the number of processes to schedule (you can assume that this number is always positive, there is no need to check for that in your program).
2. For each process to schedule, ask the user to enter the burst time for the process (you can assume that all burst times are positive, there is no need to check for that in your program).
3. Asks the user to enter the length of the time quantum q that must be used by the Round Robin algorithm (you can assume that this number is always strictly positive, there is no need to check for that in your program).

Here is an example of what running the program must look like (where **3**, **24**, **3**, **3**, and **4** are inputs from keyboard)

============================================================

Enter the number of processes to schedule: **3**

Enter the burst time of process 1: **24**

Enter the burst time of process 2: **3**

Enter the burst time of process 3: **3**

Enter the length of the time quantum q: **4**

Schedule sequence: 1 1 1 1 2 2 2 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

AWT: 5.666667

ATT: 15.666667

============================================================

**Task 3**

Modify hands-on exercise Two Task 3. Write a program with name **schedule.c**

1. It can let user choose the algorithm (RR or SJF) to use.
2. This program will let user choose an algorithm and run executable files produced in task 1 and task 2 according to the choice.
3. The program can go on if the use wants to schedule another sequence of processes.

Here is an output examples.

Example:

============================================================

Choose algorithm (rr for Round Robin, sjf for Shortest Job First, exit for exit the system): sjf

Enter the number of processes to schedule: 4

Enter the burst time of process 1: 6

Enter the burst time of process 2: 3

Enter the burst time of process 3: 1

Enter the burst time of process 4: 7

Schedule sequence: 3 2 1 4

AWT: 3.750000

ATT: 8.000000

Choose algorithm (rr for Round Robin, sjf for Shortest Job First, exit for exit the system): rr

Enter the number of processes to schedule: 3

Enter the burst time of process 1: 24

Enter the burst time of process 2: 3

Enter the burst time of process 3: 3

Enter the length of the time quantum q: 4

Schedule sequence: 1 1 1 1 2 2 2 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

AWT: 5.666667

ATT: 15.666667