**CS5250 Assignment 4**

**Lin wei**

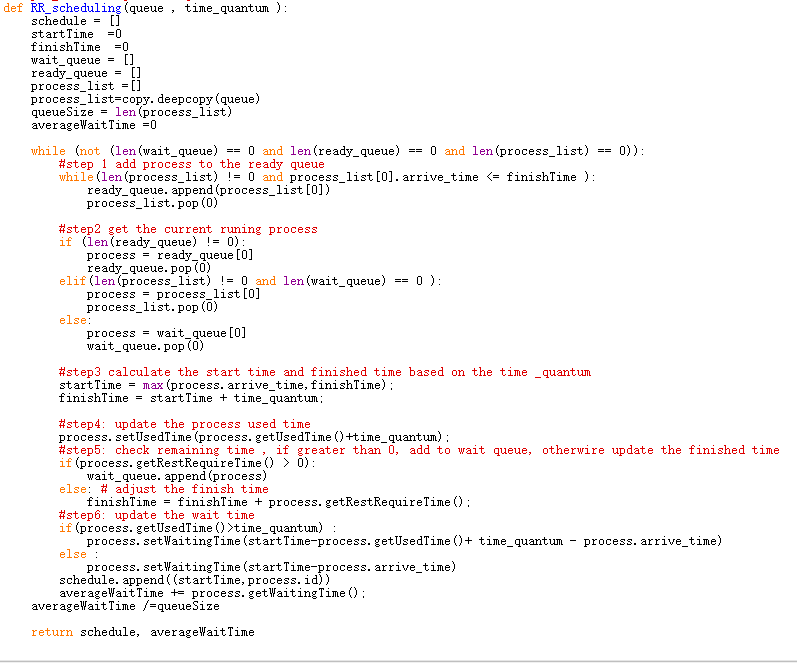
**A0110144A**

**Task 1:**

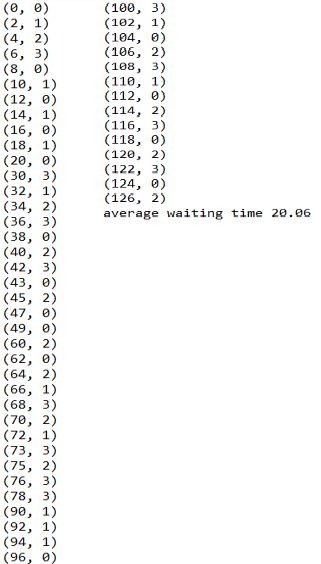
**The code and output can be found and in the following link**

**<https://github.com/aslin1126/CS5250/tree/master/assignment4>**

**1.1 Round Robin (RR)**

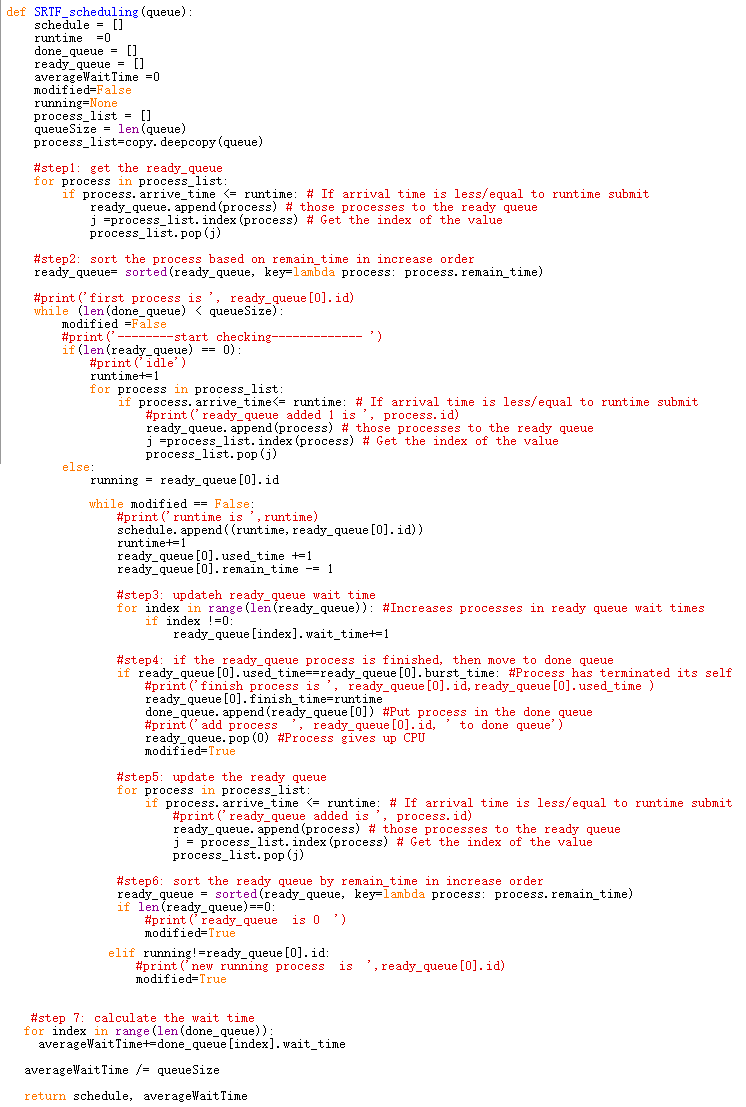
**Below shows the code implementation**

**The result for RR\_scheduling with time\_quantum =2 is showing below**

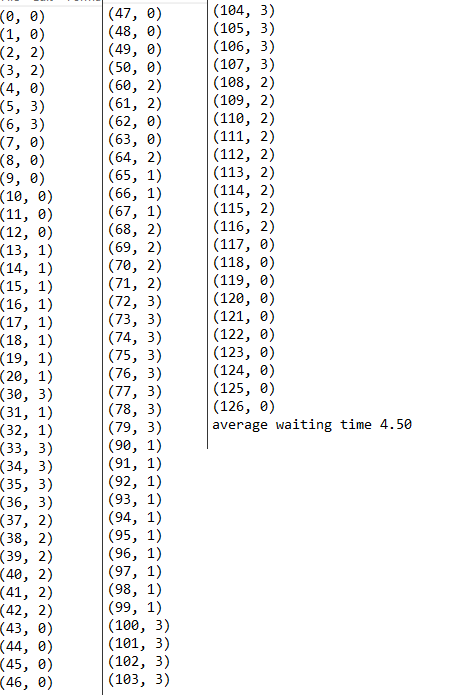


1. **2 Shortest remaining time first (SRTF) with given cpu burst time**

**Below shows the code implementation**

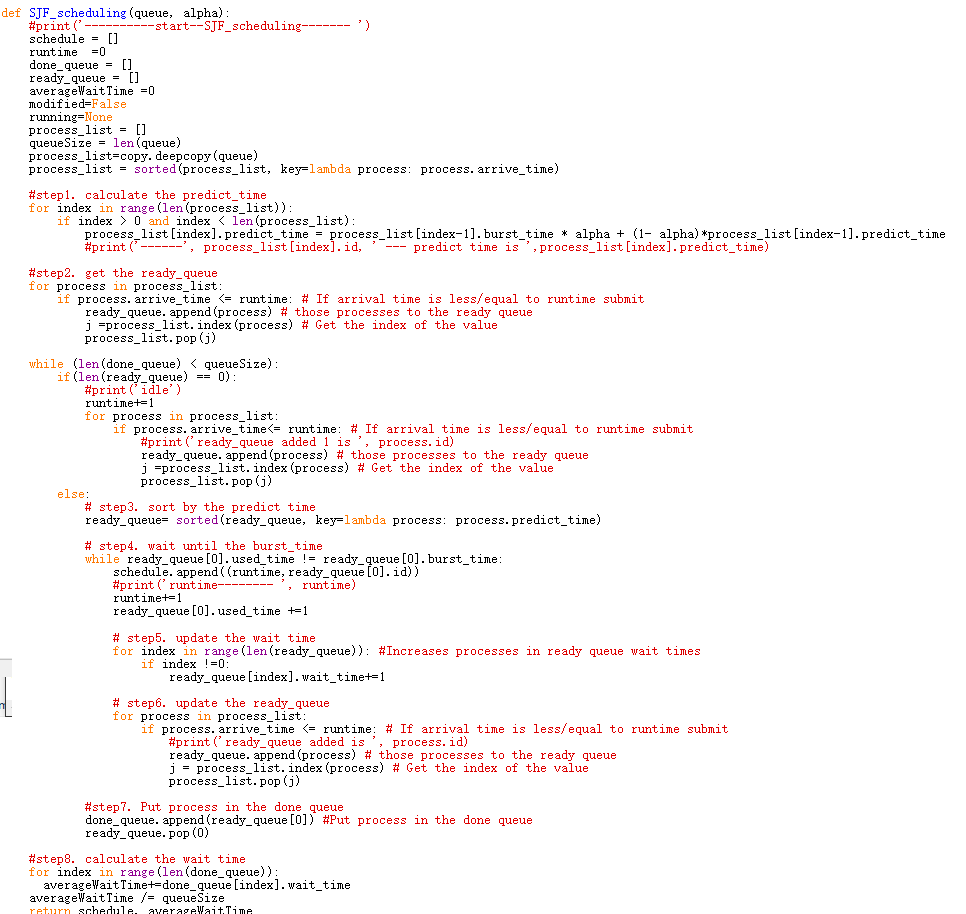


**The result for SRTF\_scheduling with time\_quantum =2 is showing below**

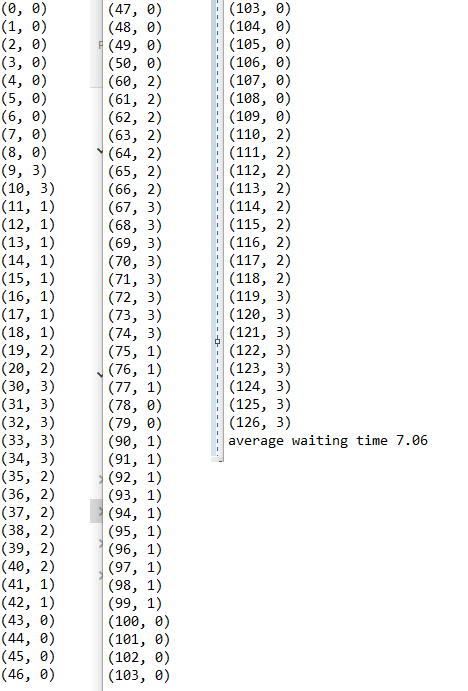


**1.3 Shortest job first (SJF) with future prediction**

**Below shows the code implementation**



**The result for SJF\_scheduling with alpha= 0.5 is showing below**



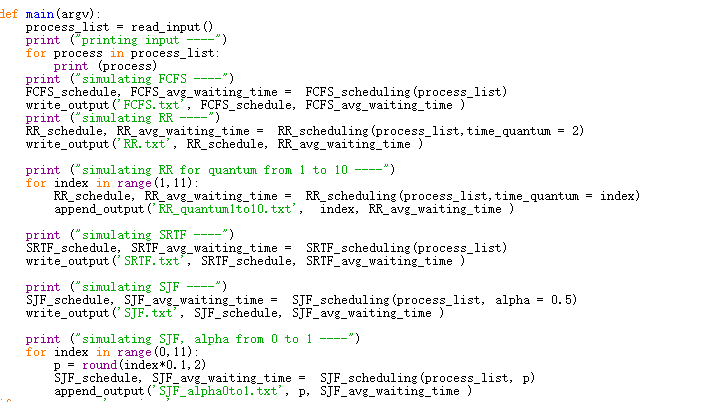
**Task 2:**

**2.1 Test with 2 of your implemented scheduling schemes (in Task 1.1 and Task 1.3) and compare them. Which one give the least average\_waiting\_time? Try to adjust the time quantum Q for RR and α value for SJF, give the optimal value of Q and α to minimize average waiting\_time for the above particular input. Include the output of your tests in the report. (4 marks)**

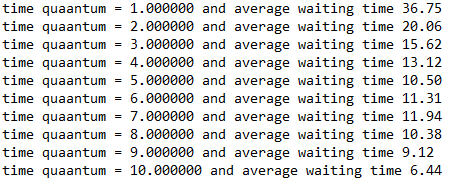
The average\_waiting\_time for RR is 20.06, while for SJF with prediction is 7.06,

Therefor, the SJF with future prediction has the least average\_waiting\_time.

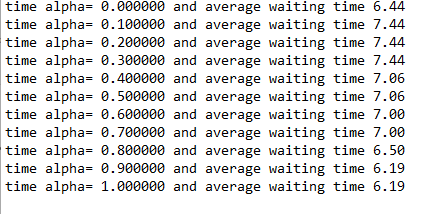
Add two more experiments to the code(shown below), first is change the time\_quantum from 1 to 10 and out put file to RR\_quantum1to10.txt. Second is change the alpha from 0 to 10 with step value 0.1 and output file to the SJF\_alpha0to1.txt



The result is shown below



For RR, when the time quantum =10, the scheduler provide the lease average waiting time which is 6.44

z

For SJF with prediction, when the alpha >=0.9, the scheduler provide the lease average waiting time

**2.2 What is the optimal scheduling scheme (gives minimum average waiting time) for a system with (4 marks):**

**a) All short processes**

FCFS works best when compared to SJF and RR, because the processes are short which means that no process will wait for a longer time, When each process is executed one by one, every process will be executed eventually.

**b) Very short and very long processes interleave each other with unpredictable pattern.**

Round Robin scheduling works efficiently here because it does not cause starvation and also gives equal time quantum for each process.

**2.3. Assume your system has N CPU cores, and each process only requires burst time on 1 core. Will it make the scheduler more complicated? Suggest how to extend the current scheduler to multi-processor system. (bonus marks)**

If there is dependence between the processes, then the scheduling will be more complicated for multi-processer system as the scheduler need to handle the correct sequence between processes with dependence. If there is not dependence between the processes, then it will be much easier. For all the scheduler mentioned above, the scheduling process will be same, the only different step is the assignment of the process to the CPU core. Assign the first N processes from the available to be handled queue (if the number of processes that ready to be handled is less then N, just assign all these available process) to the N CPU cores, Whenever the cpu core finish its process, then the scheduler will assign it with new available process.