ProcSch-V1.0

Build a simulator ProcSch-V1.0 to showcase the working of process scheduling algorithms including FCFS, Priority (Static), SRTN, and HRRN. Your implementation should be based on our class discussion. The simulation of all the above algorithms should be made based on the input and output parameters as discussed before. All necessary asssumptions and scenarios are applicable as has been indicated during our classroom sessions.

Identify all the necessary test cases to demonstrate the working of these algorithms to the TAs, and also show the comparative performance of these algorithms for a given test case to make suitable conclusions.

As of this particular simulator, we assume that all the processes considered are hypothetical, and that none of them are neither existing in the system nor have been created using fork() calls. The process details have been provided through an input file as mentioned below.

The input parameters should be clearly specified through a single file input.txt. For every process, there exists a single row/line in the input file. Every row/line comprises of (process name/id, arrival time, service time, priority) in order. Sample input.txt is provided as follows.

```
A 0 3 1
B 1 6 0
C 2 4 2
D 3 7 3
E 4 3 1
```

Reading the first row/line indicates that for a process A, arrival time is 0 with a service time of 3 units and that it has a static priority of 1. Rows/lines may be added/removed depending on the number of processes you require in your test case. The above sample input.txt shows 5 processes.

A output computed by your implemented simulator should be generated in separate files for each of the scheduling policies. For example, fcfs.txt should showcase the working of FCFS policy, srtn.txt should display the working of SRTN policy, and so on. The corresponding output for the above input.txt should have the following information in each of the output files fcfs.txt, srtn.txt, priority.txt and hrrn.txt.

```
A <Initial WT> <Intermediate WT> <FT> <TAT>
B <Initial WT> <Intermediate WT> <FT> <TAT>
C <Initial WT> <Intermediate WT> <FT> <TAT>
D <Initial WT> <Intermediate WT> <FT> <TAT>
E <Initial WT> <Intermediate WT> <FT> <TAT>

Mean WT: _____________
Mean TAT: ____________

Schedule:____________________________________________________________
```

Please ensure to include all necessary assumptions through your comments in the code. The implementation should be done either using C/C++.

As a part of your implementation, you should create four functions namely fcfs(), priority(), srtn()

and hrrn(). Each of these should be invoked from the main() function. Also, two additional functions input() for reading the input file and output() for writing output file should be created.

The typical structure of your code should be as follows.

```
fcfs()
{
/* Implementation of FCFS policy involving computations & updating
necessary data structures */

output(output-filename);  /* Write in output-filename fcfs.txt as
available post computation */
}

srtn()
{
/* Implementation of SRTN policy involving computations & updating
necessary data structures */

output(output-filename);  /* Write in output-filename srtn.txt as
available post computation */
}

priority()
{
/*   Implementation   of   Priority(static)   policy   involving
computations & updating necessary data structures */

output(output-filename);  /* Write in output-filename priority.txt
as available post computation */
}

hrrn()
{
/* Implementation of HRRN policy involving computations & updating
necessary data structures */

output(output-filename);  /* Write in output-filename hrrn.txt as
available post computation */
}

main()
{
     input();  /* Read input.txt */

     fcfs();
     srtn();
     priority();
     hrrn();
}
```

The above is a broad sturcutre for your code. You are free to make necessary changes as needed during your implementation.