Assignment #3

Due Date: December 10, 2022

Total: 100 marks

Choose only two of the following three programs.

1. (60 marks) Implement the readers writers problem using processes, where shared data is a structure as follows:

   ```
   struct {
   int datalength;
   char info[datalength];
   } INFO;
   ```

   After each operation read or write, each process will suspend itself, and place itself in a queue. A monitor program will generate random operations (either read and write) and will also monitor the scheduling queue. The monitor must ensure that two processes will not enter their critical section at the same time, and if more processes can run simultaneously, they will all be scheduled to run.

   (a) creating appropriate processes: 20 marks

   (b) solving conflicts: 20 marks

   (c) process scheduling using the monitor queue: 20 marks

   (d) testing, comments and demo of an execution: 5 marks

   Additional relevant material can be consulted in: `Sync` Examples.

   Note:

   - Please be aware that structure `INFO` cannot be declared in C in the above form (we'll get a compiler error) and your task is to find a solution for declaring and using variable shared data size.
   - We assume a maximum length `MaxLen` for shared data.

2. (60 marks) You are given a dependency relation of the following form: For each process P_i we have a list of resources that it depends on: For example if process P_i depends on process P_j and queue Q_k we will have a list of the form: $P_1 : P_j, Q_k$. Write a simple program to determine deadlocks, or more precisely if your dependency graph has cycles.

3. (60 marks) Write a CPU scheduler using the First Come First Serve Scheduling algorithm (non-preemptive, for both the CPU and I/O). Your scheduler gets its process data from a text file in a standard format (see below). You can assume that your system has a single CPU and only one I/O device (and that device operates sequentially, serving only one process at a time).

   For example:

```
1  0   100  2  200  3  25  −99...
2  10   15  2   15  2  15    2  15  2  15  2  15  2  15  2  15  2  15  2  15  2  15    2  15  2  15  2  15  −99
3  20   15  2   10  2  10    2  10  2  10  1  10  −99
```

   Notes:

   - The lines mean that we have:
     - PID =1, arrival time = 0, CPU burst= 100, I/O burst = 2, CPU burst=200, I/O burst=3, CPU burst = 25.
     - PID =2, arrival time = 10, CPU burst= 15, I/O burst = 2, CPU burst=15, I/O burst=2, CPU burst = 15, I/O burst=2, CPU burst = 15, I/O burst=2, CPU burst = 15, I/O burst=2, CPU burst = 15, I/O burst=2, CPU burst = 15, I/O burst=2, CPU burst = 15, I/O burst=2, CPU burst = 15, I/O burst=2, CPU burst = 15, I/O burst=2, CPU burst = 15, I/O burst=2, CPU burst = 15, I/O burst=2, CPU burst = 15, I/O burst=2, CPU burst = 15.
     - PID =3, arrival time = 20, CPU burst= 15, I/O burst = 2, CPU burst=10, I/O burst=2, CPU burst = 10, I/O burst=2, CPU burst= 10, I/O burst = 2, CPU burst=10, I/O burst=2, CPU burst = 10.
   - Each line ends with -99.
   - A process can contain any number of CPU or I/O bursts.
   - It should always start and end with a CPU burst.
   - You can also assume that arrival times are in non-decreasing order.

   Output:

   - For each process, compute and display the turnaround time and the waiting time.
   - For the entire system, compute and display the average turnaround time, average waiting time, and throughput.

   For the input above, we have:

   - Total CPU burst for P1 is 100+200+25=325, for P2 is 14*15=210, and for P3 is 15+5*10=65.
   - CPU utilization is (325+210+65)/(495+(2+15)*7)=(600)/(614)=97.71 % (until 495 is 100%).

- Turn around time for 1st process is: (380-0)=380 Turn around time for all three processes is 495+119=614. For process two is 614-10=604, for process three is 480-20=460. The average turnaround time is (380+604+460)/3=481.
- Throughput is the total number of jobs completed/total time required for their execution, 3 in 614.
- Adding the second line (and others), some processes must be put in a waiting queue, thus, waiting time will increase from 0 to a number that you'll have to compute using your program.

  For example: for P1 we have a waiting time of (130-102)+(355-333)=50 and for P3 (115-20)+(345-132)+(395-357)+13*2+14=386, while for P2 the waiting time is (100-10)+(330-117)+(380-347)+8*4=368. So, the average waiting time is (50+385+368)/3=268.

Marking scheme:

(a) receiving tasks and creating the CPU queue (15 marks).

(b) creating the I/O queue (15 marks).

(c) activating/suspending tasks at the right time(10 marks).

(d) computing final results and producing the output (10 marks).

(e) a demonstration of your programs (5 marks).

(f) Overall design of your program (5 marks).

CPU queue:

0-100	100-115	115-130	130-330	330-345	345-355	355-380	380-395	395-405		404-420	420-430	430-445	445-455	455-470	470-480	480-495	497-510	512-527	...	599-614
P1	P2	P3	P1	P2	P3	P1	P2	P3		P2	P3	P2	P3	P2	P3	P2	P2	P2	P2	P2

I/O queue:

100-102	115-117	130-132	330-333	345-347	355-357	395-397	405-407	420-422		430-432	445-447	455-456	470-472	495-497	...	597-599
P1	P2	P3	P1	P2	P3	P2	P3	P2		P3	P2	P3	P2	P2	P2	P2