Department of Computer Science
University of Saskatchewan
CMPT 332
Midterm Exam

---

| | |
|---|---|
| October 30, 2013 | Name:_____ |
| Number of Pages: 5 | Signature:_____ |
| Time: 50 minutes | Student Number: _____ |
| Total: 50 marks. | NSID: _____ |

**CAUTION** - Candidates suspected of any of the following, or similar, dishonest practices shall be dismissed from the examination and shall be liable to disciplinary action.

1. Having at the place of writing any communication devices, any books, papers or memoranda, calculators, audio or visual cassette players, or other memory aid devices.
2. Speaking or communicating with other candidates.
3. Purposely exposing written papers to the view of other candidates. The plea of accident or forgetfulness shall not be received.

| Q1 | Q2 | Q3 | Q4 | Q5 | Q6 | Q7 | Q8 | TOTAL |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| 5 | 6 | 5 | 12 | 10 | 4 | 3 | 5 | 50 |

1. (5 marks) Assume you have 640 KB of main memory. Processes A, B, C, D, E and F with their associated memory requirements are presented to the medium-term scheduler for execution in the following order. How is memory allocated for the following sequence of operations for loading these 6 processes into memory using the Worst Fit algorithm? Show your answer by providing one diagram of RAM **after** all the allocation and deallocation has been completed.

| Process | Size (in KB) |
|---|---|
| A | 180 |
| B | 100 |
| C | 244 |
| D | 98 |
| E | 240 |
| F | 70 |

Operations to perform:

- Allocate D,
- Allocate A,
- Allocate C,
- Free A,
- Allocate E,
- Allocate B,
- Allocate F,
- Free D.

IF the operations cannot be completed in the given order, indicate which operations would have to be aborted. Do not complete an aborted operation at a later time.

2. (6 marks). An operating system usually provides a method of preventing a user program from directly accessing protected memory space and from executing a certain set of assembly instructions. Describe (in as much detail as the page permits) the general method for executing a **system call**. Only describe the run-time behaviour, **not** what is necessary to **program** a system call.

3. (5 marks). For the following Cobegin/CoEnd code, draw the Process Flow Graph and give the Fork/Join code that corresponds to that particular code sequence.

```
COBEGIN
  P1
  P2
  BEGIN
    P3
    COBEGIN
      P4
      P9
    COEND
    P5
  END
COEND
```

4. (12 marks) Process Scheduling. Consider the following job mix, including relative arrival times.

| Job | Arrival Time | CPU Burst Time |
|---|---|---|
| A | 0 | 10 |
| B | 3 | 8 |
| C | 4 | 8 |
| D | 5 | 4 |
| E | 7 | 5 |

Show the order of execution of these jobs using Round-Robin with a quantum of 3 and Shortest Remaining Time First. Use a time-line like in the interactive examples and/or class showing which jobs execute during which time slots. For RR, jobs are assumed to arrive just **before** the time slot indicated and are placed at the **back** of the ready queue.

```
RR

-------------------------------------------------------------------------
|        |
-------------------------------------------------------------------------
 0      3  etc...

TAT:                                          EFF:
```

```
SRTF

-------------------------------------------------------------------------
|        |
-------------------------------------------------------------------------
 0

TAT:                                          EFF:
```

Indicate the avg. turnaround time (TAT) and the **specific CPU efficiency** for each algorithm on this set of tasks. For the efficiency, assume that each context switch takes .2 of a time unit. For the timeline, assume that each context switch takes 0 time. No calculator is necessary; leave the efficiency as a fraction.

5. (10 marks). Semaphores, Monitors, and Rendezvous IPC can all solve the same classes of problems, in that you can implement one facility with one of the others. Use Monitors to implement the IPC primitives Send, Receive and Reply as defined in class to have zero-capacity semantics. Provide detailed **C-like pseudocode**. Identify the List Operations that you need (if you need any), but do not implement them. Assume that you have a monitor facility in your programming language, so that you do not need MonEnter and MonLeave, and that messages consist of a pointer to shared space and a buffer length. Assume also that the monitor contains the PCB for all the processes. Define those fields in the PCB which you think are necessary (in particular, condition variables). I've got you started. The implementation is short.

```
Monitor IPC
{
/*shared  data and initialization */
```

```
/* Monitor procedures  */
char *Send(
{
```

```
}
```

```
char *Receive(
{
```

```
}
```

```
int Reply (
{
```

```
}
```

6. (4 marks) List 1 (one) major motivation from the operating system view for having multiple execution cores on the same chip. What two advantages do multiple cores have over multiple processors? What major disadvantage do they have over multiple processors?

7. (3 marks) What is the major negative consequence of a quantum that is too short in uniprocessor Round Robin scheduling? What is the consequence of a quantum that is too long?

8. (5 marks) You've been asked to design and implement a communication protocol. The protocol will be used to support communication for a server your company uses. You ask your boss

   *"Does the server in this communication protocol provide an idempotent or a non-idempotent service?"*

   Your boss gives you a look like you're nuts and asks you what you are talking about. Please define the meaning of the word idempotent in this context, and explain why it is important for you to know this given the task at hand.

——————————- THE END——————————