1. Consider the following solution to the mutual exclusion problem:

```c
#define FALSE 0
#define TRUE 1

shared int reserved[2] = {FALSE, FALSE};

void enter_region(int pid) {
    while (reserved[other]); // busy wait
    reserved[pid] = TRUE;
} // enter_region

void leave_region(int pid) {
    reserved[pid] = FALSE;
} // leave_region
```

a) What is wrong with it? (5 points) **It does not guarantee mutual exclusion.**

b) When does this problem happen? (5 points) **When two processes perform enter_region in lockstep.**

2. Consider the following System V Release 4 scheduler:

<table>
<thead>
<tr>
<th>#ts_quantum</th>
<th>ts_tqexp</th>
<th>ts_slpret</th>
<th>ts_maxwait</th>
<th>ts_lwait</th>
<th>LEVEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>800</td>
<td>0</td>
<td>1</td>
<td>16000</td>
<td>0</td>
<td># 0</td>
</tr>
<tr>
<td>400</td>
<td>0</td>
<td>2</td>
<td>8000</td>
<td>2</td>
<td># 1</td>
</tr>
<tr>
<td>200</td>
<td>2</td>
<td>3</td>
<td>4000</td>
<td>3</td>
<td># 2</td>
</tr>
<tr>
<td>100</td>
<td>2</td>
<td>4</td>
<td>2000</td>
<td>4</td>
<td># 3</td>
</tr>
</tbody>
</table>

and identify the four incorrect parameters:

a) Parameter _ts_tqexp_ on level _2_ should not be equal to _2_.

b) Parameter _ts_slpret_ on level _3_ should not be equal to _4_.

c) Parameter _ts_lwait_ on level _0_ should not be equal to _0_.

d) Parameter _ts_lwait_ on level _3_ should not be equal to _4_.

3. Give two examples of data structures that are much costlier to pass to a remote procedure call than to a regular procedure call (2×5 points)

a) First example: __a large array__

b) Second example: __a long linked list__

4. What does the VMS scheduling policy do to favor interactive processes over other processes? (10 points)

They increase the priorities of processes reading from or reading to the terminal by a higher increment than those of processes doing disk I/Os. In addition, the reward for doing a read from a terminal is higher than the reward for doing a write to a terminal.
5. A restaurant has a dining room that can seat a maximum of 40 guests. Complete the following code fragments to ensure (a) that the room will never contain more than 40 guests and (b) that parties will either have all their guests seated or all their guests waiting. (20 points).

```c
semaphore chairs = __40__;
semaphore access = __1__;
enter_room (int nguests) {
    int i;
    __P(&access__);  
    for (i = 0; i < nguests ; i++)
             __P(&chairs__);  
    __V(&access__);  
} // enter_room
leave_room (int nguests) {
    int i;
    for (i = 0; i < nguests ; i++)
             __V(&chairs__);  
} // leave_room
```

6. Answer in one sentence or less to each of the following questions (6×5 points)
   a) What is the major disadvantage of busy waits?
      They result in too many context switches and, more generally, waste CPU cycles.
   b) What is the major disadvantage of atomic transactions?
      They are costly to implement.
   c) What is the major advantage of datagrams over virtual circuits?
      They have less overhead because they do not require the establishment of a connection between the sender and the receiver.
   d) How can we implement the at-most-once semantics in remote procedure calls?
      By attaching a serial number to each request and instructing the server to reject duplicate requests.
   e) When should we worry about big-endians and little-endians?
      Whenever we are transmitting data between two different machines as their byte ordering could be different.
   f) What is the difference between reliable datagrams and unreliable datagrams?
      Reliable datagrams are acknowledged by their receiver.