Max marks = 12
Total time = 60 minutes

Question 1. A barbershop consists of a waiting room with n chairs, and the barber room containing the barber chair. If there are no customers to be served, the barber goes to sleep. If a customer enters the barbershop and all chairs are occupied, then the customer leaves the shop. If the barber is busy, but chairs are available, then the customer sits in one of the free chairs. If the barber is asleep, the customer wakes up the barber. Write a program to coordinate the barber and the customers. [4 marks]

To make the problem a little more concrete, there is additional information:

1. Customer threads should invoke a function named getHairCut.
2. If a customer thread arrives when the shop is full, it can invoke balk, which does not return.
3. The barber thread should invoke cutHair.
4. When the barber invokes cutHair there should be exactly one thread invoking getHairCut concurrently.

Write a solution that guarantees these constraints using semaphores. Write the pseudocode for the barber and the customers.

Hint: you may want to use the following variables. You can choose to ignore this hint too and work out an alternative solution.

1. N – total number of customers that the shop can accommodate (N-1 waiting in the wait room, and 1 getting a hair cut)
2. Customers – to count number of customers in the shop
3. Some mutex variable for …
4. Barber and Customer variable
5. CustomerDone and BarberDone variables

Question 2. What are the 4 conditions for deadlock [1 mark]

Question 3. Why is it a bad idea to disable interrupts to build a lock? [1 mark]

Question 4. The following code defines a function that inserts a node with an integer value at the head of the list.

```
1   void insert(int value) {
2     node_t *n = malloc(sizeof(node_t));
3     assert(n != NULL);
4     n->value = value;
5     n->next  = head;
6     head     = n;
7   }
```

Why will this function not be thread safe? [1 mark]

Using the CompareAndSwap atomic instruction whose software realization is as follows, modify the above function to make it thread safe. [1 mark]

```
1   int CompareAndSwap(int *address, int expected, int new) {
2     if (*address == expected) {
3       *address = new;
4       return 1; // success
5     }
6     return 0; // failure
7   }
```

Question 5. Write pseudocode using condition variables where the parent thread waits for the child thread to finish before it resumes executing [1 mark]

Question 6. Students in the dining hall invoke dine and then leave. After invoking dine and before invoking leave a student is considered "ready to leave". The synchronization constraint that applies to students is that, in order to maintain the illusion of social suave, a student may never sit at a table alone. A student is considered to be sitting alone if everyone else who has invoked dine invokes leave before she has finished dine. Using semaphores write pseudocode that enforces this constraint. [3 marks]