1. **(20 points)** Suppose that each search request for Google search engine takes 5 ms CPU time for processing and a total of 15 ms to read/write the information from/to the database on a disk. Moreover, there is only one CPU and a single disk.

   What are the throughputs (in terms of requests/second) for the following situations? 1 second is equal to 1000ms.

   Explain the assumptions that you have and the reasoning for your results.

   - Single-threaded implementation of Google server;
   - Multi-threaded implementation and how many threads are needed?

   Suppose that Google needs to provide service at the level of 2,000 search requests/second, how can you design the search engine to achieve such a performance objective (number of threads, CPUs and disks needed)?

2. **(15 points)** Suppose that two threads A and B are sharing one variable ‘account’, which has the initial value of 100; Now, thread A is going to make a deposit and perform: ‘account = account + 200’, while thread B is going to do a withdraw: ‘account = account – 50’.

   What are the possible values of the variable after these operations? Assuming that we don’t have correct synchronizations on these two operations. Context switches can happen at any point.

3. **(15 points)** Suppose that we have the atomic hardware instruction “TestAndSet(a)”.

   Show how to use this instruction in threads A and B to solve the problem in (2). List your pseudo code.

4. **(15 points)** Suppose that there are three threads A, B and C, where each of them has two sections of codes C1 and C2; the execution of 2nd section of code C2 in threads A and B need to wait the completion of the 1st section of code C1 in C. Use semaphores to solve the synchronization problem. You need to define and initialize the semaphores. List your pseudo code.

5. **(15 points)** Use conditional variables to solve the synchronization problem listed in Question 4. List your pseudo code.

6. **(20 points)** Revisit the Peterson’s Solution for synchronization, explain why it satisfies the mutual exclusion, progress, and bounded waiting separately.