1. You are constructing the database for the ATO and have decided to use a B+-tree data structure where the internal nodes contain dummy keys and the keys which point to the taxpayer records are located at the terminal nodes. Each key consists of a 9-digit TFN. Each node of the B+-tree will occupy one 4Kbyte disk block. To locate both a disk block a 32-bit disk address and 16-bit disk device address are needed. The access time for each disk block is 2.5ms and the access time for a 4-byte word of memory is 150ns. Assume there are 18x10^6 taxpayers.
   (a) How much space will you use to store the indices of the database?
   (b) How long will it take to find a taxpayers records?

2. Consider the following Hexadecimal keys:
   11 83 62 53 99 23 78 54 31 33
   (a) Calculate the values of the hash function h(k) = floor(a*k) mod M for each of the following cases and indicate the number of collisions:
      (i) a = 1, M = 16
      (ii) a = 1, M = 13
      (iii) a= 0.618, M = 16
   (b) Compare the number of collisions for each case in (a) with the chosen values for a and M. What can conclusions can you draw? Are your conclusions generally applicable to all possible lists of keys? Explain!

3. (a) How long could it take in the worst case to insert N keys into an initially empty table using:
   (i) separate chaining with unordered lists?
   (ii) separate chaining with ordered lists?
   (iii) linear probing?
   (b) How much additional space is needed in the worst case to insert N keys into initially empty table for each case in (a)?

4. Give the contents of the has table that results when you insert items with the keys E A S Y Q U T I O N in that order into an initially empty table of M = 5 lists, using separate chaining with unordered lists. Use the hash function 11k mod M to transform the k\textsuperscript{th} letter of the alphabet into a table index.