1. A hash table has 5 buckets and uses the hashing function h(*i*) = (2*i*+3) mod 5. Collisions are handled by chaining. Draw the hash table resulting from hashing the following keys:

   12, 3, 9, 11, 19, 21, 4, 8.

2. Consider the problem of searching for genes in DNA sequences using Horspool's algorithm. A DNA sequence consists of a text on the alphabet {A, C, G, T} and the gene or gene segment is the pattern.

   Note: Horspool's algorithm is given on the last page of this exam.

   a. Construct the shift table for the following gene segment of chromosome X:

      ATCTCCTA

   b. Apply Horspool's algorithm to locate the above pattern in the following DNA sequence. Indicate the position of the search string after each iteration of the algorithm.

      ATCTAGATCTCGTATTTTTTATAGATCTCCTAT

3. Step through a heapsort to sort the following array H=[28, 21, 25, 18, 9, 7, 13]. Note that the array is a max heap. Show the array H after each readjustment of the heap.

4. Consider the algorithm shown below. This is a rather obscure sorting algorithm known as comb sort. It is easy to implement, and rivals the speed of quicksort on many common input sets. The basic idea in this algorithm is to reduce the number of swaps for small values near the end of the list.

   a. Identify the basic operation in the algorithm.
   b. In the worst case, how many times is the basic operation executed?
   c. What is the worst-case efficiency class of this algorithm?

```
algorithm combsort(A[0..n-1])
    gap ← A.size - 1
    while gap ≠ 1 and swaps ≠ 0
        if gap > 1
            gap ← gap / 1.3
            if gap = 10 or gap = 9
                gap ← 11
            end if
        end if
        i ← 0
        swaps ← 0
        while i + gap < A.size
            if A[i] > A[i+gap]
                swap(A[i], A[i+gap])
                swaps ← swaps + 1
            end if
            i ← i + 1
        end loop
    end loop
end algorithm
```

5. Let A = {a_1,..., a_n} and B = {b_1, ..., b_m} be two sets of numbers. Consider the problem of finding their intersection, i.e., the set C of all the numbers that are in both A and B.
   a. Provide detailed pseudocode for an algorithm to solve this problem in better than $O(n^2)$ time.
   b. Determine the efficiency class of your algorithm.

6. An auto parts store verifies and updates their inventory every year. Luckily everything in the store is barcoded. All they have to do is walk around with a barcode scanner and scan each item that is on a shelf or in the warehouse. You have been asked to write a program that will report the quantity of each item that is in stock.

   Input to your program is from two files. The first file contains a list of the *n* inventory items the store sells, along with the corresponding barcode. The second file contains a list of *m* barcodes that were scanned.

   Output is a new file containing one line for each inventory item, along with the quantity currently in stock. The output is sorted by decreasing quantity on hand.

   Sample Input

```
File 1: part_descriptions.txt
Bumper 101224
Headlight 223433
Manifold 123456
Solenoid 656565
Tire 001998
```

```
File 2: scanned_codes.txt
101224
223433
001998
223433
223433
123456
101224
```

   Sample Output

```
File 3: quantity_on_hand.txt
Headlight 3
Bumper 2
Manifold 1
Tire 1
Solenoid 0
```

   a. Outline (in English or very high level pseudocode) an efficient algorithm to solve this problem. Be sure to list any major data structures that are needed, as well as how they will be used.

   b. What do you expect the worst case efficiency of your solution to be? Justify your answer.