**Exercises 3.2:**

5.

How many comparisons (both successful and unsuccessful) are made by the brute-force string matching algorithm in searching for each of the following patterns in the binary text of 1000 zeros?

 a) 00001 The Search String (0000….000) is of length 1000; the Search Pattern (00001) is of length 5. Hence, there will be 1000-5+1 = 996 iterations. In each of these iterations, the first 4 comparisons would be successful and the last comparison will be unsuccessful. Hence, there will be 996\*4 = 3,984 successful comparisons and 996\*1 = 996 unsuccessful comparisons. Total comparisons = 3984 + 96 = 4980 comparisons.

b) 10000 There will be a total of 1000-5+1 = 996 iterations. In each of these iterations, the first comparison would itself be unsuccessful. Hence, there will be 996\*1 = 996 unsuccessful comparisons and there will not be any successful comparisons. Total comparisons = 996.

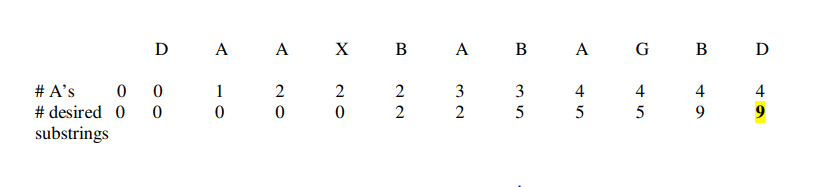
c) 01010 There will be a total of 1000-5+1 = 996 iterations. In each of these iterations, the first comparison would be successful and the second comparison would be unsuccessful. Hence, there will be 996\*1 = 996 successful comparisons and another 996\*1 = 996 unsuccessful comparisons. Total comparisons = 1992.

8.

Consider the problem of counting, in a given text, the number of substrings that start with an A and end with a B. (For example, there are 9 such substrings in DAAXBABAGBD). Design a Θ(n) algorithm to count such substrings.

Note that the number of desired substrings that end with a B at a given position i (0 < i ≤ n-1) in the text is equal to the number of A’s to the left of that position. This leads to the following algorithm:

Initialize the number of A’s encountered and the number of desired substrings encountered to 0. Scan the text from left to right. When an A is encountered, increment the number of A’s encountered. When a B is encountered, increment the number of desired substrings encountered by the current value of the number of A’s encountered. When the text is exhausted, return the last value of the number of substrings encountered. Since, we do a linear pass on the text and spends constant time on each of its characters, the algorithm is linear.

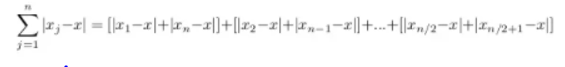


**Exercises 3.3**

3.

a)If we put the post office at location xi, the average distance between it and all the points x1<x2<...<xn is given by the formula 1/n |xj-xi|. Since the number if points n stays the same, we can ignore the multiple 1/n and minimize We will have to consider the cases of even and odd n separately.

Let n be even. Consider first the case of n = 2. Thu sum |x1-x| + |x2-x| is equal to x2-x1 , the length of the interval with the endpoints at x1 and x2, for any point x of this interval (including the end points), and it is larger than x2-x1 for any point x outside the interval. This implies that for any even n1 the sum



is minimized when x belongs to the interval . If x must be one of the points given either x (n/2) or x (n/2+1) solves the problem.

Let n>1 be odd, Then, the sum is minimized when x == x [n/2], the point for which the number of the given pointts to the left of it is equal to the number of the given points to the right of it.

Note that the point smallest called the median - solves the problem for even n’s as well. For a sorted list implemented as an array, the median can be found in 0(1) times by simply returning the [n/2]th.element of the array.

b) Assuming that the points x1,x2,x3,...,xn are given in increasing order, the answer is the point xi, that is the closest to m = (x1+xn) /2, the middle point between x1 and xn. (The middle point would be the obvious solution if the post office did not have to be at one of the given locations.) Indeed, if we put the post office at any location xi, to the left of m, the longest distance from a village to the post office would be x,-xi, this distance is minimal for the rightmost among such points. If we put the post office at any location xi, to the right of m, the longest distance from a village to the post office would be xi-x1: this distance is minimal for the left most among such point

